Ph.D. DISSERTATION

VARIATION-AWARE DESIGN AND PACKAGING TECHNIQUES IN 3D ICS

공정변이를 고려한 3차원 집적 회로 설계 및 패키징 기법

BY

Sangdo Park

FEBRUARY 2014

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Ph.D. DISSERTATION

VARIATION-AWARE DESIGN AND PACKAGING TECHNIQUES IN 3D ICS

공정변이를 고려한 3차원 집적 회로 설계 및 패키징 기법

BY

Sangdo Park

FEBRUARY 2014

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

VARIATION-AWARE DESIGN AND PACKAGING TECHNIQUES IN 3D ICS

공정변이를 고려한 3차원 집적 회로 설계 및 패키징 기법

지도교수 김 태 환

이 논문을 공학박사 학위논문으로 제출함

2013년 11월

서울대학교 대학원

전기컴퓨터 공학부

박 상 도

박상도의 공학박사 학위 논문을 인준함

2013년 12월

위 원 장: _____

부위원장: _____

위 원: _____

위 원: _____

위 원: _____

Abstract

As CMOS scaling down, The control of variation in chip performance (i.e. speed and power) becomes highly important to improve the chip yield. The increased variation of chip performance demands additional design efforts such as the increase of guard-band or longer design turnaround time (TAT), which cause degradation of both chip performance and economic profit. Meanwhile, through-silicon via (TSV) based 3D technology has been regarded as the promising solution for long interconnect wire and huge die size problem. Since a 3D IC is manufactured by stacking multiple dies which are fabricated in different wafers, integration of the dies that have far different process characteristic can enlarge the difference of device performance on different dies within a single chip. In this dissertation, we analyze the effect of on-package (within-chip) variation on 3D IC and presents effective methods to mitigate the on-package variation. First, a parametric yield improvement method is presented to resolve the mismatches of dies having different process characteristic. Comprehensive 3D integration algorithms considering post-silicon tuning technique is developed for the multi-layered 3D IC. Then, we show that a careful clock edge embedding in 3D clock tree can greatly reduce the impact of on-package variation on 3D clock skew and propose a two-step solution for the problem of on-package variation-aware layer embedding in 3D clock tree synthesis. In summary, this dissertation presents effective 3D integration method and 3D clock tree synthesis algorithm for process-variation tolerant 3D IC designs.

keywords: VLSI&CAD, 3-dimensional IC, clock tree synthesis, process variation, post-silicon tuning

student number: 2009-30915

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Process Variation in 3D ICs

As CMOS scaling down, the impact of process variation has been severe and becomes a major concern for circuit design because it increases the uncertainty of key electrical parameters (i.e. speed and power) of devices and interconnect of an integrated circuit. The increased variation of chip performance, especially speed and power, demands additional design efforts such as the increase of guard-band or longer design turnaround time (TAT), which cause degradation of both chip performance and economic profit. Predictive technology models expect that the magnitude of process variation will be further increased in the future CMOS generations. For example, the magnitude of variation (σ/μ) in inverter delay is predicted to increase from almost 4% in 45nm technology to 25% in a 12nm technology [7], where σ and μ are the standard deviation and mean of inverter delay, respectively. Therefore, it is essential to take the effect of the process variation into consideration in design stage.

The process variation can be classified into two categories: die-to-die (D2D) and within-die (WID) variation. The D2D variation is the collective effect of lot-to-lot,

wafer-to-wafer variations, which is fully correlated for a given die, but separate dies can have different amount of variation. On the other hand, WID variation correspond to the variability of process parameters within a single chip, which is location-dependent. For example, devices located in close to each other are more likely to have the similar characteristic than those placed far away. In general, it is known that D2D variation can dominated WID variation. However, in recent years, WID variation is rapidly growing and can significantly affect the variability of performance parameters on a chip [8]. Meanwhile, through-silicon via based 3D IC design has been regarded as one of doable solutions to alleviate the long interconnection and huge die size [9,10]. A 3D stacked IC is made by splitting a large design into several dies and stacking these dies into a package. Unlike 2D ICs, these dies assembled in a 3D package might exhibit different process corners because the dies are manufactured in different lots and wafers with having various process characteristics. Therefore, both WID and D2D variation in 2D chips contributes on-package variation in 3D chips. In spite of the different variation effect in 3D chips, generally, the variability of performance in 3D IC is smaller than in 2D IC due to the smaller die area and the averaging effect of variation on data path [1,11–13]. Fig. 1.1 shows the process corner distributions for 64-bit adder in four configurations in [1]. From the result, it is clearly shown that the variability of chip performance is tightening in both leakage and timing as the number of layers in the 3D stack increases. However, integration of the dies that have far different process corners can also enlarge the difference of device performance on different dies within a single chip. For example, 3D clock network clock skew can experience unintended huge clock skew which is defined as the difference of propagation delays from source to sinks [2,6,14]. Fig. 1.2 illustrates the comparison of 2D (the propagation delay difference between two points in the same die) and 3D (the propagation delay difference between two points in the different dies) clock skew distribution of specific point in the clock networks [2]. In Fig. 1.2, the all clock network types show that the variation

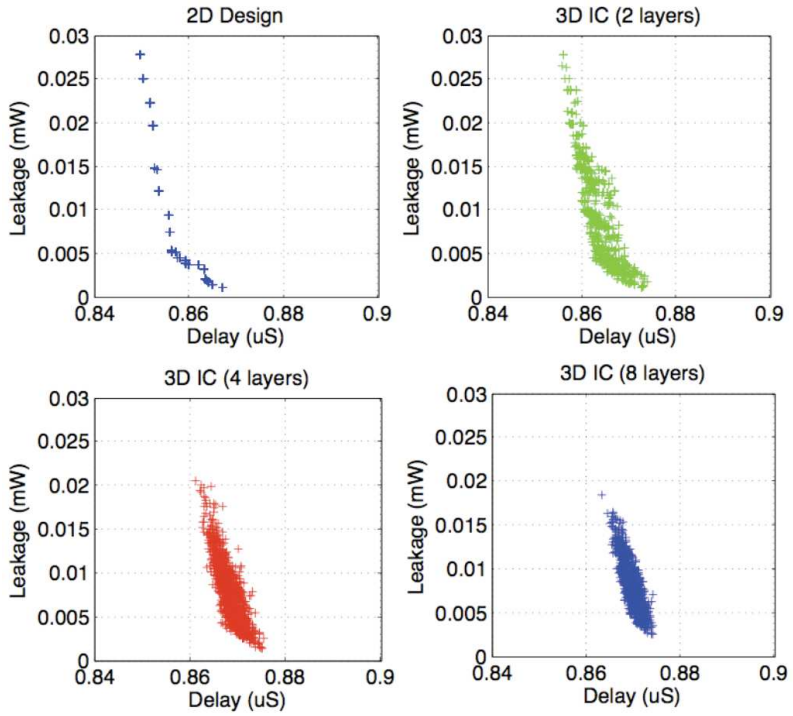
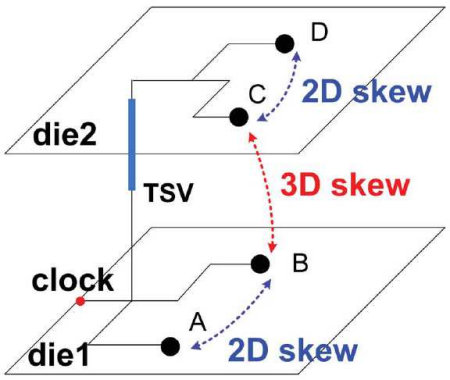
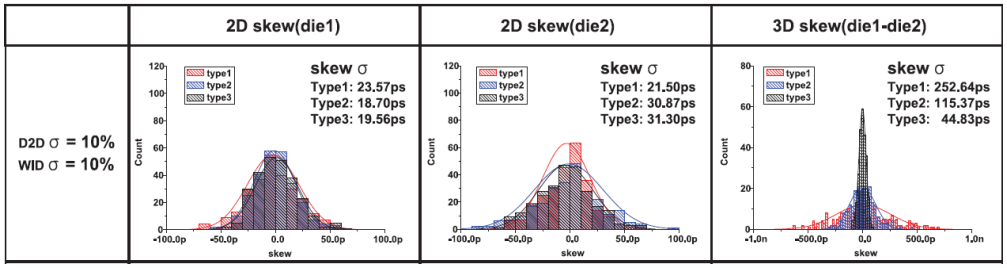


Figure 1.1: Process corner distributions for 64-bit adder in four configurations under 90nm library [1]; (a) a traditional planar configuration ; (b) 2-layer 3D stack; (b) 4-layered 3D stack (c) 8-layered 3D stack.



(a)



(b)

Figure 1.2: 2D and 3D skew distributions of specific points in the clock network according to different variation [2].

in 3D clock skew is extremely higher than the variation in in 2-D clock skew. Since the clock paths in 3D clock network are divided into different dies, D2D variation contributes to worsening clock skew variation, whereas clock skew variation in 2D clock network is determined by only WID variation. Further, variations in RC properties of through-silicon-vias (TSVs) is also the source of propagation delay variation in 3D ICs [15]. Therefore, it is necessary to develop methodologies to mitigate the delay and power spread of 3D chips considering on-package variation.

The works that have addressed the problem of on-package variation aware yield enhancement in 3D ICs can be classified into three categories: (1) wafer-to-wafer, die-to-wafer, and die-to-die matching techniques (e.g., [3, 4, 16]); (2) post-silicon tuning techniques (e.g., [3,17,18]); (3) delay path distribution techniques (e.g., [6,13–15,17]); Reda, Smith, and Smith [4] proposed a set of wafer-to-wafer matching algorithms to maximize the functional yield of 3D stacked ICs, while Ferri *et al.* [16] presented die-to-die and die-to-wafer matching algorithms to maximize the parametric yield. Recently, Kim and Kim [3] proposed a die-to-die matching algorithm with the consideration of body biasing as a post-silicon tuning to enhance the parametric yield. However, there exists an achievable range that the die matching algorithm can mitigate the on-package variation because its concern is only the elimination of mismatches of dies having different process corners. On the other hand, Chae and Mukhopadhyay [17] proposed a level shifter insertion method with the tier adaptive voltage scaling as post-silicon tuning. They attempted to explore all the discrete insertion candidates of each level shifter in 3D paths and placed the shifter to the location that was likely to enable to minimize the delay variation of 3D paths. However, the problem of finding exact location at which the level shifters are placed to minimize the delay variation of critical path in a circuit has not been fully investigated. Recently, Kim, Joo, and Kim [18] proposed adjustable delay buffer (ADB) insertion method to reduce clock skew variation in multiple supply voltage designs. Even though the use of ADBs is effective in

reducing clock skew variation, the work is confined to 2D IC designs.

1.2 Contributions of This Dissertation

For the mitigating the variability of chip performance in 3D IC design, this dissertation present core algorithms and optimization techniques for 3D clock network. Chapter 2 analyze the on-package variation (OPV) and effect of post-silicon tuning method on general case of 3D clock network, and present a novel die/wafer matching algorithms for 3D clock skew. Then, a cost effective 3D CTS algorithm for highly tolerant to the OPV variation induced timing variation is studied in Chapter 3. Finally, Chapter 4 concludes this dissertation. The contributions of this dissertation are summarized as follows:

• **Chapter 2** analyzes the limitations of previous post silicon management methods for the reduction of OPV induced 3D clock skew in 3D stacked IC designs. Then, a subsequent-matching aware die/wafer matching algorithms are presented to mitigate the OPV induced 3D clock skew. Through extensive experiments, it is shown that it can effectively enhance parametric yield for all 3D-integration types (i.e. die-to-die, die-to-wafer and wafer-to-wafer integration).

• **Chapter 3** addresses the problem of on-package variation aware layer embedding in 3D clock tree synthesis and present two-step solution, LEE-CLK, to mitigate the on-package variation on 3D clock skew, which leads high chip yield. From the extensive experiments, the proposed algorithm is able to improve the chip yield efficiently.

Chapter 2

Post-silicon Tuning Aware Die/Wafer Matching Algorithms for Enhancing Parametric Yield of 3D IC Design

For enhancing parametric yield of clock tree in in through-silicon via (TSV) based 3D IC designs, this chapter proposed die-to-die/die-to-wafer/wafer-to-wafer matching algorithms.

2.1 Introduction

Through-silicon via (TSV) based 3D IC design has been regarded as one of the doable solution to reduce wire length and footprint of a chip [9, 10]. However, the timing closure problem (e.g., meeting timing or clock period constraint) in 3D IC has shown that it is much more complicated and unpredictable than in 2D IC because timing variability in 3D IC is contributed by die-to-die (D2D) as well as within-die (WID) variation [1, 11–13]. In general, 3D chips are made by splitting a large design into several dies

and stacking these dies into a package. Unlike 2D ICs, these dies assembled in a 3D package might exhibit different process corners because the dies are manufactured in different lots and wafers with having various process characteristics. Therefore, Integration of the dies that have far different process corners can enlarge the difference of propagation delays on paths across multiple dies.

Clock skew, defined as the difference of propagation delays from source to sinks, is one of important design parameters. Since unintended clock skew incurs the system failure or performance degradation, many techniques for clock tree synthesis are targeted for Elmore delay based zero skew or certain skew bound [19, 20]. However, the actual clock skew after fabrication stage can't be the same as the target clock skew in design stage under the influence of process variation induced timing variability. In addition, in 3D clock network, D2D variation also contributes to worsening clock skew variation, whereas clock skew variation in 2D clock network is determined by only WID variation. Therefore, it is necessary to develop design methodologies to minimize clock skew variation considering within-chip variation on 3D clock network.

In the area of 2D clock tree synthesis, an extensive effort has been made to mitigate the clock skew variation (e.g., [21–23]). A common feature among many of the variation-aware 2D clock tree synthesis methods is that they try to find proper (x, y) location of clock tree nodes by integrating Statistical Static Timing Analysis (SSTA) into Deferred-Merge Embedding (DME) algorithm [19]. However, these works in 2D clock tree synthesis inherently can't be applied to 3D clock network and don't take some of circuit elements working at far different process corners into account. Recently, in the 3D clock tree synthesis flow, a number of works have actively researched on finding proper layer assignment of clock tree for mitigating the skew variability [6, 14, 15]. Hu *et al.* [14] demonstrated empirically under the use of 3D H-tree structure that the skew variation between a pair of clock sinks can be reduced if the sinks and their connecting edges are all together placed in the same layer. Zhao *et al.* [6] further elaborated

the idea in that they aggressively attempted to route the most part of clock paths to the same layer at the expense of the increase of TSV usage. In addition, Yang *et al.* [15] proposed an on-package variation aware 3D clock tree synthesis methodology to reduce the clock signal variation. The idea is to place the clock buffers in clock tree on as many uncorrelated dies as possible during the process of clock tree synthesis. Even though these works in [6] and [15] are effective in mitigating the effect of on-package variation on the clock tree, their methods have limitation to reduce D2D variation in 3D stacked ICs and they demand much more TSVs than necessary.

To resolve the mismatches of dies having different process corners in 3D ICs, there have been extensive works in post-silicon stage: (1) wafer-to-wafer, die-to-wafer, and die-to-die matching techniques (e.g., [3,4,16]); (2) post-silicon tuning techniques (e.g., [3, 17]); wafer-to-wafer, wafer-to-die, and die-to-die matching techniques are 3D integration methodology to select wafers (or dies) to form 3D chip for the improvement of functional yield, parametric yield, or some profits [4,16,24].[1] Reda, and Smith [4] proposed a set of wafer-to-wafer matching algorithms to maximize the functional yield of 3D stacked ICs. Ferri *et al.* [16] presented die-to-die and die-to-wafer matching algorithms to maximize the parametric yield for a CPU-to-L2 cache die stack, considering the variation of operation frequency of CPU and access latency of L2 cache. Common drawback of the previous algorithms in [4, 16] is that (drawback-1) for 3D ICs with $K(> 2)$ layers, as the value of K increases, the quality of solution is far from the optimum, and (drawback-2) no post-silicon tuning method is taken into account in the die matching process, which results in low parametric yield. Kim and Kim [3] proposed a die-to-die matching algorithm considering body biasing as a post-silicon tuning technique to increase the parametric yield [3]. However, the work is limited to two-layered

[1] *Functional yield* refers to the percentage of chips manufactured that are functionally correct while *parametric yield* refers to the percentage of chips manufactured under process variation that are correct in terms of timing, power, etc.

3D ICs only even though it resolves drawback-2. In this work, we propose a die-to-die , die-to-wafer and wafer-to-wafer 3D integration strategies using post-silicon tuning techniques for mitigating 3D clock skew variation. Precisely, the contributions of the work are the followings:

- We comprehensively improve the die-to-die and die-to-wafer algorithm by [3], so that our proposed algorithm is well applicable to 3D ICs with $K(> 2)$ layers while it overcomes the drawback-1 as well as drawback-2;
- We extend die-to-die algorithm to wafer-to-wafer matching method for improving parametric yield considering functional yield and post-silicon tuning method simultaneously.

The remainder of this paper is organized as follows. In section 2.2, the review of the previous works that are used for die-to-die, die-to-wafer and wafer-to-wafer matching algorithms using post-silicon method is presented. Then, we provide a brief overview of motivational idea and formulate the main problem of improving the yield of die-to-die and die-to-wafer integration in section 2.3. In section 2.3, we propose a solution to consider post-silicon tuning more effectively for die-to-die, die-to-wafer 3D integration. Then, we extend our die-to-die solution to the wafer-to-wafer matching problem in section 2.4. Section 2.5 provides a comprehensive set of experimental result and conclusions that demonstrate the effectiveness of our proposed approaches.

2.2 Preliminaries

Since our die-to-die matching algorithm for 3D ICs with $K(\geq 2)$ layers is based on the algorithm in [3] which works only for two-layered ICs, a description on the algorithm in [3] is briefly given in the following. Let us consider the problem of matching dies in two die sets where one set contains the upper dies when stacked and the other

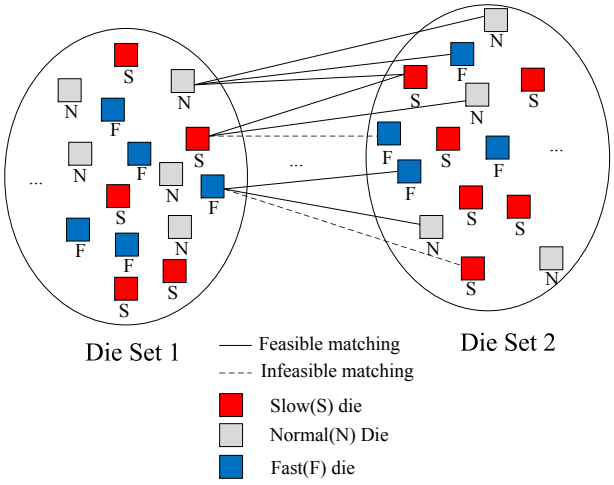


Figure 2.1: Examples of feasible (solid line) and infeasible (dashed line) die-to-die matchings when body biasing is used as a post-silicon tuning technique.

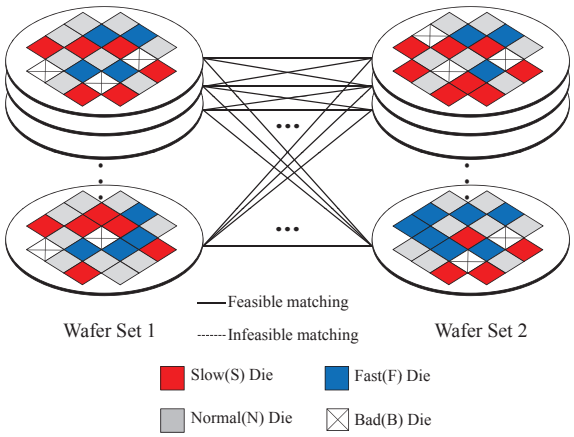


Figure 2.2: Examples of wafer-to-wafer matchings where wafers contain dies characterized as slow, fast, normal and bad.

set contains the lower dies when stacked. We assume that each die in the sets has been characterized (in terms of timing in this example) and classified into one of the three process corners: slow, fast, and normal[2] In addition, we assume that a slow (fast) die in a set can be stacked with a slow (fast) or a normal die in the other set without timing violation by applying a proper post-silicon tuning technique such as body biasing to the dies, but stacking a fast (slow) die to a slow (fast) die causes timing violation whatever post-silicon tuning techniques are applied. We call a matching (i.e., stacking) of two dies *infeasible* if the matching incurs timing violation that can never be resolved with any intended post-silicon tuning technique. Otherwise, the matching is called *feasible*.[3] Fig. 2.1 shows examples of feasible and infeasible die-to-die matchings when body biasing is used as post-silicon tuning technique, in which it is assumed that no body biasing is able to fix the timing violation when a slow die is matched with a fast die. For wafer-to-wafer matching, Fig. 2.2 illustrates an examples of wafer-to-wafer matchings where the wafer set on left side contains the upper wafers and the other set contains the lower wafers when they are stacked. Basically, wafer-to-wafer matching share the common elements with die-to-die matching such as the corner classified dies and pass/fail criteria to determine *feasible/infeasible* matching between the stacked dies. However, since the locations of dies are pre-fixed in wafers, the result of wafer matching can be expressed as not binary information (*feasible/infeasible*) but numerical information such as the sum of *feasible* matchings in dies belong to the wafers. In addition, since there exist functionally defective dies in wafers unlike only good dies in die-to-die matching, the previous works in wafer-to-wafer matching try to avoid stacking good and bad die together for improving functional yield [4, 16].

[2]For the ease of presentation, only three process corners are shown in this example. Actually, 21 process corners are used in the experiments

[3]Note that the feasibility and infeasibility of die matching follows according to the classification of process corners and body biasing voltage range. In addition, the definition of feasibility and infeasibility can be similarly extended to the case of more than two dies to be stacked.

2.3 The Die-to-Die Matching Problem and Proposed Algorithm Considering Body Biasing

2.3.1 Motivation and Problem Definition

First, we want to illustrate, using a simple example, what critical limitation the previous die matching method [3] has. Fig. 2.3(a) shows a sequence of die matching used by [3] for stacking three dies, in which the dies in the first column are maximally matched with the dies in the second column, producing four (feasible) matches (d_1, d_5), (d_2, d_7), (d_3, d_6) and (d_4, d_8) and then, the resulting stacked dies are matched with the dies in the third column maximally, producing three (feasible) matches (i.e., 75% yield) as shown in the heavy lines in Fig. 3.2(a). Note that matching (d_3, d_6) with either d_{11} or d_{12} is infeasible because body biasing cannot resolve the timing violation at all when the slow die d_6 in (d_3, d_6) is stacked on the fast dies d_{11} or d_{12}. On the other hand, Fig. 3.2(b) shows another feasible matching result for the same sequence of die matching. The die matching between the first and second columns produces a set of four matches that is different from that in Fig. 2.3(a). Subsequently, the resulting stacked dies are maximally matched with the dies in the third column, producing one more feasible matches (i.e., 100% yield) than that in Fig. 3.2(a). The reason that the matching algorithm in [3] produces an inferior result is due to the complete unawareness of the effect of the current matching on the result of the succeeding matchings in the matching sequence. Our proposed new matching algorithm, as presented in the next section, overcomes this limitation.

The die-to-die feasible matching problem can be described as:

Die-to-die matching (DDM) problem: *Given K die sets $\{D_1, \ldots, D_K\}$ where each die set D_i, $i = 1, \ldots, K$, consists of N dies $d_1^i, \ldots, d_N^i$ such that their process corners have been identified by wafer level testing and the body biasing voltage range*

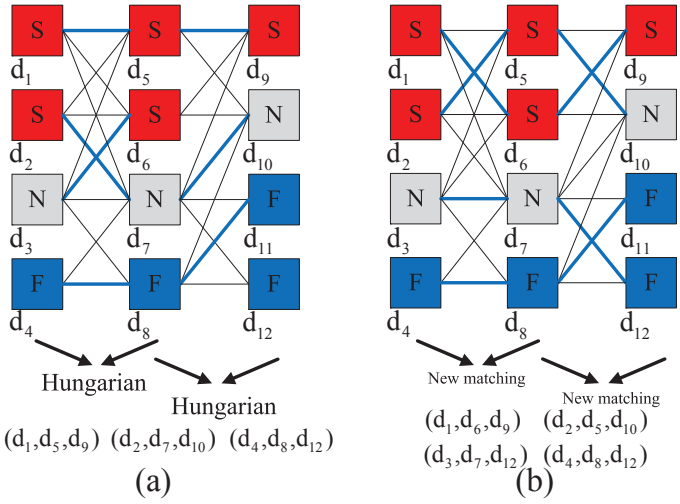


Figure 2.3: Limitation of previous die matching method; (a) the result produced by using [3], achieving 75% parametric yield, (b) another matching result that achieves 100% parametric yield.

($V_{BB.MIN}$ $V_{BB.MAX}$), find a maximal K-dimensional feasible matching from the sets $D_1, \ldots, D_K$.

The problem is equivalent to the traditional K-dimensional matching problem if all the feasible K-dimensional matches are initially given, the number of which is essentially bounded exponentially. Our approach is different from the traditional K-dimensional matching solution, which basically examines all feasible matches exhaustively, in that we want to find a maximal K-dimensional matching by constructing K-dimensional matches step-by-step by iteratively applying our 'subsequent-matching aware' two-dimensional matching algorithm that is proposed in the following section.

2.3.2 The Proposed Die-to-Die Matching Algorithm

Given that the process corners of all dies in K sets $D_1, \ldots, D_K$ are identified by the wafer-level testing and the body biasing voltage range ($V_{BB.MIN}$ $V_{BB.MAX}$), we iteratively perform two-dimensional matching; firstly performing die-to-die matching between the dies in D_1 and D_2, then, die-to-die matching between the 2-die set obtained from the previous iteration and D_3, ..., lastly performing die-to-die matching between the $(K-1)$-die set obtained from the previous iteration and D_K. For each iteration of die matching between $(i-1)$-die set and D_i, $i = 2, \ldots, K-1$, we construct a bipartite graph in which the nodes in the left column represent the $(i-1)$-stacked dies in the $(i-1)$-die set and the nodes in the right column represent the dies in D_i. We put an arc from every node in the left column to every node in the right column. For the ease of presentation, we assume to use three process corners of S (slow), N (normal), and F (fast). Then, we define one decision variable $X(d_v, d_w)$ between two dies d_v and d_w such that $X(d_v, d_w)$ = 1 if matching (stacking) d_v with d_w is feasible under the body biasing voltage range. Otherwise, $X(d_v, d_w)$ = 0. Let $L(p, q)$ denote the set of i dies that correspond to the $i-1$ dies in node p of the left column and one die in node q of the right column. Then, we measure the degree of (potential) matching feasibility

between p and q by using parameter $\rho(p, q)$:

$$\rho(p,q) = \prod_{d_i \in L(p,q)} X(d_S, d_i) + \prod_{d_i \in L(p,q)} X(d_N, d_i) + \prod_{d_i \in L(p,q)} X(d_F, d_i) \quad (2.1)$$

where d_S, d_N, and d_F represent dies of slow, normal, and fast process corners, respectively. Note that $X(d_S, d_F) = X(d_F, d_S) = 1$, and for the rest of $X(\cdot, \cdot)$, the values are 0.

Each of the first, second, and third product terms in Eq. 2.1 will be set to 1 if every die in $L(p, q)$ to be stacked is feasible in matching with d_S, d_N, and d_F, respectively. Thus, the higher the value of $\rho(p, q)$ is, the higher the possibility of subsequent feasible matching will be. Let M be the number of process corners. (In our presentation, M = 3.) We assign the value of $M - \rho(p, q)$ to the arc from p to q in the bipartite graph. Note that $\rho(p, q)$ = 0 means that stacking all the dies in p and q are infeasible. We remove the infeasible arcs from the bipartite graph. Finally, we add two nodes called *source* s and *sink* t, as shown in Fig. 3.3 and create arcs from source to all nodes in the left column and arcs from all nodes in the right column to sink. We assign a cost of 0 to every arc connected from source or to sink. The capacity of every arc is set to 1. Then, our die-to-die matching problem at each iteration is translated into the problem of finding a maximum flow of minimum cost in the network, which is solvable efficiently in polynomial time [25, 26].

Figs. 2.4(a) and (b) show the construction of networks for solving the sequence of two 2-dimensional die matching problem instances in Fig. 2.3(b) and the derivation of maximum flow of minimum cost in the networks. For example, for the arc from node d_3 to d_8 in Fig. 3.3(a) where d_3 and d_8 are labelled N and F, respectively, its assigned cost is $M - \rho(d_3, d_8)$ = 3 - 2 = 1 since $\rho(d_3, d_8)$ = 0 + 1 + 1 in Eq. 2.1. For the arc from (2-die stacked) node $d_2 \cdot d_3$ to d_9 in Fig. 3.3(b) where d_2, d_3, and d_9 all are labelled with S, its assigned cost is $M - \rho(d_2 \cdot d_5, d_9)$ = 3 - 2 = 1 since $\rho(d_2 \cdot d_5, d_9)$ = 1 + 1 + 0 = 2 in Eq. 2.1. From the flow in the network in Fig. 3.3(a) four 2-die matchings

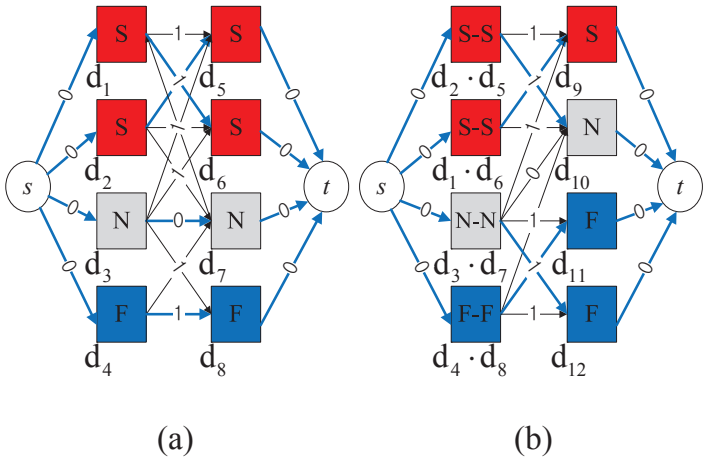


Figure 2.4: The construction of networks for solving the sequence of two 2-dimensional die matching problem instances and the derivation of maximum flow of minimum cost in the networks; (a) the network for the first 2-dimensional matching, (b) the network for the second 2-dimensional matching

$(d_2 \cdot d_5)$, $(d_1 \cdot d_6)$, $(d_3 \cdot d_7)$ and $(d_4 \cdot d_8)$ are derived. Then, from the flow in Fig. 3.1(b) four 3-die matchings $(d_2 \cdot d_5 \cdot d_{10})$, $(d_1 \cdot d_6 \cdot d_9)$, $(d_3 \cdot d_7 \cdot d_{12})$, and $(d_4 \cdot d_8 \cdot d_{11})$ are finally derived.

2.4 The Wafer-to-Wafer Matching Problem and Proposed Algorithm Considering Body Biasing

2.4.1 Problem Definition and The Proposed Wafer-to-Wafer Matching Algorithm

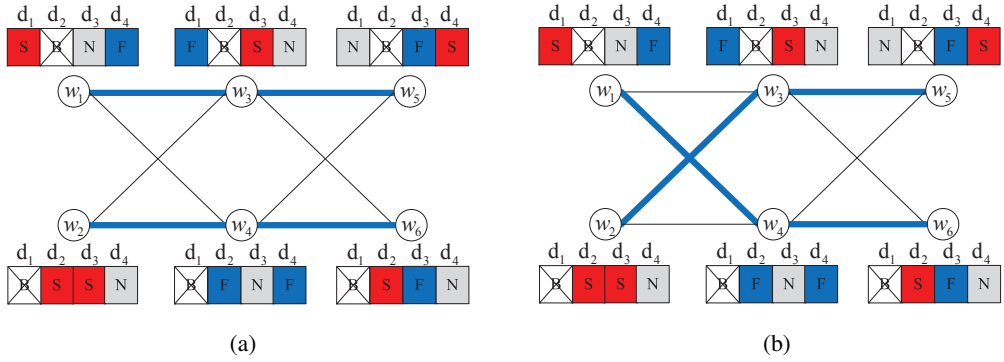


Figure 2.5: Limitation of the previous wafer matching method; (a) the result produced by using [4], achieving 12.5% parametric yield, (b) another matching result that achieves 50% parametric yield.

In this section, we show the limitation of the previous wafer matching method [4] and extend our "subsequent-matching aware" die matching algorithm to the wafer matching algorithm. Fig. 2.5 shows the comparison of two different matching result produced by the previous method [4] which tries the optimal 2-D wafer matching by changing the order of matching sequence and the other method which achieve higher parametric yield (e.g. 50%). Even though the previous method achieves the optimal 2-D W2W matching for the maximum parametric yield (e.g. 50%) in first column and

second column, the final parametric yield (e.g. 12.5%) is much lower than that(e.g. 50%) of the wafer matching result shown in Fig. 2.5.(b). The reason that the previous method produce the inferior result is also due to the unawareness of subsequent matching similar to the previous die matching algorithm in section 2.3. However, since a wafer in W2W matching contains a set of corner identified dies as well as a set of functionally bad dies which incur system failure, we need an additional consideration including the effect of bad dies which turn any stacked dies which has high matching feasibility into bad package. Before we introduce our proposed algorithm, we formulate the W2W matching problem formally. In wafer-to-wafer matching, the die set D_i in wafer w_i can be represented as a string of a process corner (e.g. S for slow corner, and F for fast corner) or a 0 which indicate a bad die identified by known good die(KGD) test. If two wafers with die set D_i and D_j are integrated to form 3D IC, the resultant die set $D_i \cdot D_j$ is also formed as a string of stacked process corner (e.g. $S \cdot N$) or a 0 which indicates infeasible matching between two good dies or matching between a good die and a bad die.

The wafer-to-wafer matching problem can be formulated as : **Wafer-to-wafer matching (WWM) problem**: *Given K wafer sets $\{W_1, \ldots, W_K\}$ where each wafer set W_i, $i = 1, \ldots, K$, consists of M wafers $w_1^i, \ldots, w_M^i$ such that the process corners and defects in a die set D_j have been identified by level testing and known good die testing, respectively, and the body biasing voltage range ($V_{BB.MIN}$ $V_{BB.MAX}$) is given, find a maximal K-dimensional feasible matching from the sets $W_1, \ldots, W_K$.*

The problem is also equivalent to the traditional K-dimensional matching problem as in [4]. We propose a heuristic similar to the proposed die-to-die matching algorithm in Section 2.3.2. We iteratively perform two-dimensional(2D) matching from the wafers in W_1 and W_2 to the $(K-1)$-wafer set obtained from the previous iteration and D_K. For each iteration of 2D wafer matching, we construct a complete bipartite graph in which the nodes in the left column represent the $(i-1)$-stacked wafers in

the $(i-1)$-wafer set and the nodes in the right column represent the wafers in W_i. However, For additional constraints in wafer-to-wafer matching, the 2D die matching algorithm in Section. 2.3.2 isn't able to directly apply to 2D wafer matching. Unlike there is no restriction in the selection of two dies in die-to-die matching, the integration of two dies in wafer-to-wafer matching is pre-determined by their the (x, y) location in two wafers to be matched. Above all, The existence of the functionally defective dies(bad dies) in wafers is a critical factor to lower matching feasibility. For example, if a die which has high potential degree of feasibility defined in Eq.2.1 is matched with bad die, the result of this matching is turned to be infeasible. Therefore, we proposed a new cost function $\sigma(w_p, w_q)$ as the weight on an arc between w_p, w_q to consider simultaneously the conventional matching method to maximize functional yield and potential feasible matching using body biasing proposed in Section 2.3.2:

$$\sigma(w_p, w_q) = (1-\lambda)\sum_{i=0}^{N} Y(d_i^p, d_i^q)/N + \lambda \sum_{i=0}^{N} \rho(d_i^p, d_i^q)/(C \cdot N) : \qquad (2.2)$$

where λ, N, C, d_i^p and d_i^q represent a constant from 0 to 1, the number of die in a wafer, the total number of precess corners, i-th die in wafer W_p and W_q, respectively. Note that $Y(d_i, d_j)$ is a function that return 1 if both d_i, d_j are good die, otherwise return 0. Then, we apply Hungarian algorithm [27] for finding a matching solution to maximize weighted sum. The proposed W2W matching algorithm is described in Algorithm 1 which tries to find the best matching result by increasing λ from 0 to 1 iteratively. Note that $G(W_i)$ return the total number of feasible matching in stacked dies in a wafer set and Δ is set to 1/5.

2.5 Experimental Results

The proposed die/wafer matching algorithms are implemented using C++ and run on a Linux machine with 2.66 GHz Intel Core2 Duo processor and 2GB memory. We

Algorithm 1 subsequent-matching aware wafer-to-wafer matching algorithm

Input: K wafer set $W = \{W_1, \ldots, W_K\}$;

Output: M stacked wafers;

$\lambda = 0$, $Yield_{best} = 0$, $Yield = 0$

• $min\text{-}cut \leftarrow \infty$; // minimizing cut means maximizing sharing of layers

repeat

    $W_o = W_1$

    for $i = 2$ increase to K **do**

        Perform Hungarian(W_o, W_i, λ)

        $W_o = W_o \odot W_i$

        $\lambda = \lambda - \delta$

    end for

    $\lambda = \lambda + \Delta$ where $0 \leq \Delta \leq 1$

    $Yield = G(W_o)$

    if $Yieldld_{best} \leq Yield$ **then**

        $Yieldld_{best} = Yield$

    end if

until $\lambda \leq 1$

evaluate the performance of our matching algorithms for benchmark circuits in ISPD 2009 clock network synthesis contest [28] under Elmore delay model. 45nm predictive technology parameters [29] are used for SPICE simulation. To evaluate the effect of the reduction of skew variation on the chip yield, we define a parametric yield $Y = \sum_{i=1}^{N} x_i/N$ where $x_i = 1$ if $|skew_{max}| \leq B_{skew} + M_{skew}$ and 0, otherwise. B_{skew}, M_{skew}, $skew_{max}$ and N represent the base clock skew, the clock skew margin, the maximum clock skew, and the size of die/wafer set, respectively. We set B_{skew} as the clock skew of 3D ICs integrated with all slow process corner dies. 3D clock trees are constructed by the 3D clock tree synthesis flow with interconnect technology parameter proposed in [3]. Table. 2.1 summarizes the initial multi-layered 3D CTS result without on-package variation. The first column shows the benchmark circuits and the next three columns indicate the number of TSVs allocated, the total wire length, the propagation delay from clock source to sink, respectively. We assume that process corner of each dies are identified and normally distributed in the 21 process corners, which are discretized bin between slow/fast corner and nominal corner for 10% step. The range of body biasing is limited by the control capability of voltage regulator and the reliability characteristics of devices. In this work, we assume that we are allowed to use body biasing in the range $\pm 0.2V$. The body biasing voltages to each dies are applied by the pre-defined value for each process corner combinations in [3], which the pre-defined value are determined by using the clock buffer propagation delay trend. our algorithms, the previous method in [3] denoted Hungarian and an integer-linear programming (ILP) [16] are applied to the 3-layered, 4-layered, and 5-layered designs using the process corners identified by 100 dies ($N_D = 100$) under Gaussian distribution for D2D variation. In W2W matching, the negative binomial distribution is used to generate defect wafer maps, where the yield of an individual wafer is explored from 85% to 95% as does in [4]. Each wafer sets consists of 25 wafers and each wafer contains 1278 dies followed the experimental setup in [4]. For comparison purpose, our

Benchmark	TSVs					Wirelength(mm)					Latency(ns)				
# of layers	3	4	5	6	7	3	4	5	6	7	3	4	5	6	7
ispd09f11	51	76	95	102	123	105.1	92.2	80.9	75.3	68.7	0.404	0.394	0.344	0.354	0.293
ispd09f12	58	70	83	109	120	94.9	86.7	76.2	71.0	66.7	0.385	0.371	0.312	0.303	0.288
ispd09f21	56	74	98	115	117	111.6	97.6	85.9	79.2	73.7	0.43	0.386	0.365	0.337	0.309
ispd09f22	44	59	67	76	95	61.8	57.3	51.2	47.0	46.6	0.33	0.311	0.296	0.244	0.278
ispd09f31	130	174	221	241	280	226.7	195.8	180.6	166.0	159.5	0.598	0.514	0.464	0.446	0.443
ispd09f32	57	129	163	165	181	173.4	149.3	135.7	124.1	118.8	0.607	0.53	0.45	0.434	0.433
ispd09f33	105	148	172	187	226	172.1	151.2	134.2	128.5	115.3	0.531	0.513	0.446	0.441	0.406
ispd09f34	80	105	213	142	156	144.5	123	112.3	108.5	99.2	0.535	0.504	0.443	0.435	0.413
ispd09f35	92	137	162	177	191	161.8	138.9	127.3	120.8	110.0	0.517	0.513	0.446	0.435	0.38
ispd09fnb1	54	67	72	67	75	33.3	32.9	33.7	35.1	33.1	0.155	0.179	0.16	0.15	0.164
ispd09fnb2	140	159	168	191	206	75.1	68.9	70.0	64.2	64.0	0.248	0.212	0.217	0.216	0.211

Table 2.1: Initial multi-layered 3D CTS result

algorithms, the algorithm in [4] denoted as IMH, and an integer-linear programming (ILP) [4] are applied to the 3-layered, 4-layered, 5-layered, 6-layered and 7-layered designs.

• **Assessing the quality of our subsequent-matching aware die matching algorithm**: Table. 2.2 and 2.3 show comparisons of the yields and run times of the algorithms in [3] denoted by Hungarian, ours, and ILP for 0ps and 5ps clock skew margin, respectively. The GNU linear programming kit (LPK) is used to run the ILP formulation. Note that we are not able to apply the ILP for 4-layered and 5-layered designs because the number of variables and constraints exceeds the range that the ILP solver can handle. In summary, our die-matching algorithm is able to improve the yield by 0∼2%, 1∼7%, and 3∼8% over the previous method for 3-layered, 4-layered, and 5-layered designs, respectively while spending a little run time (less than 1 sec).

• **Comparing the quality of sequential matching sequence by ours and in [3] with**

Layers	3.0			4.0		5.0	
Stacking Method	Sequential		ILP	Sequential		Sequential	
Matching Method	Hungarian	Ours		Hungarian	Ours	Hungarian	Ours
ispd09f11	70.3	76.2	70.3	59.4	61.4	30.7	59.4
ispd09f12	100.0	100.0	100.0	99.0	100.0	100.0	100.0
ispd09f21	100.0	100.0	98.0	80.2	88.1	93.1	97.0
ispd09f22	91.1	88.1	93.1	97.0	88.1	98.0	95.1
ispd09f31	100.0	100.0	100.0	79.2	85.2	80.2	87.1
ispd09f32	92.1	93.1	94.0	74.3	89.1	100.0	100.0
ispd09f33	83.2	92.1	87.1	75.3	87.1	74.3	81.2
ispd09f34	96.0	98.0	94.1	89.1	99.0	87.1	98.0
ispd09f35	87.1	83.2	83.2	83.2	95.1	84.2	100.0
ispd09fnb1	98.0	100.0	96.0	97.0	100.0	98.0	100.0
ispd09fnb2	94.1	100.0	95.1	74.3	79.2	100.0	100.0
Average	91.99	93.70	91.89	82.54	88.39	85.96	92.53
Ratio	1.00	1.02	1.00	1.00	1.07	1.00	1.08
Run Time(ms)	1.74	24.49	253457.00	2.46	35.42	4.07	58.00
Ratio	1.00	14.05	145414.23	1.00	14.40	1.00	14.24

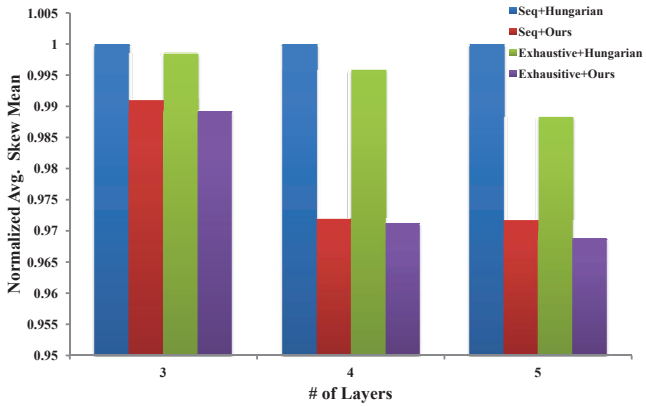
Table 2.2: Parametric yield achieved by the previous work (Hungarian), ILP and our algorithm (Ours). The clock skew margin is set to $0ps$.

Layers	3.0			4.0		5.0	
Stacking Method	Sequential		ILP	Sequential		Sequential	
Matching Method	Hungarian	Ours		Hungarian	Ours	Hungarian	Ours
ispd09f11	97.0	98.0	96.0	83.2	81.2	96.0	100.0
ispd09f12	100.0	100.0	100.0	100.0	100.0	100.0	100.0
ispd09f21	100.0	100.0	100.0	99.0	100.0	97.0	100.0
ispd09f22	97.0	94.1	100.0	100.0	100.0	99.0	100.0
ispd09f31	97.0	100.0	100.0	97.0	99.0	97.0	100.0
ispd09f32	100.0	100.0	100.0	99.0	100.0	100.0	100.0
ispd09f33	98.0	100.0	98.0	80.2	90.1	94.1	99.0
ispd09f34	100.0	100.0	100.0	99.0	100.0	98.2	100.0
ispd09f35	97.0	97.1	97.0	100.0	100.0	91.1	100.0
ispd09fnb1	100.0	100.0	100.0	100.0	100.0	99.0	100.0
ispd09fnb2	100.0	100.0	100.0	100.0	100.0	100.0	100.0
Average	98.74	99.02	99.19	96.13	97.30	97.41	99.91
Ratio	1.00	1.00	1.00	1.00	1.01	1.00	1.03

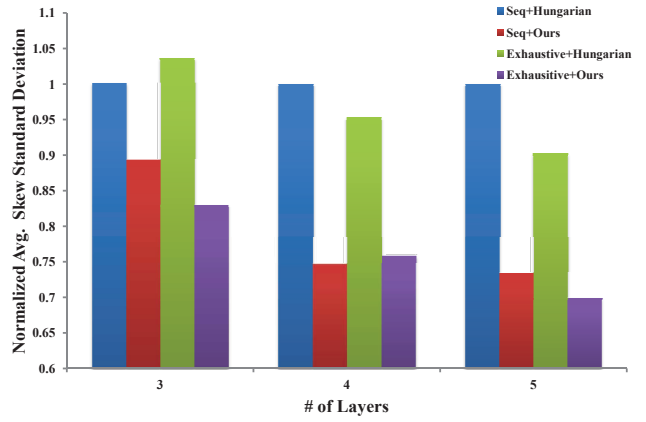
Table 2.3: Parametric yield achieved by the previous work (Hungarian), ILP and our algorithm (Ours). The clock skew margin is set to $5ps$.

	Yield (%) for #layer = 3		Yield (%) for #layer = 4		Yield (%) for #layer = 5	
	sequential	exhaustive	sequential	exhaustive	sequential	exhaustive
Hungarian	92.0	90.4 (-1.8%) 2.9x slow	82.5	83.2 (+0.8%) 17x slow	86.0	89.2 (+3.8%) 136x slow
Ours	93.7	95.1 (+1.4%) 2.7x slow	88.4	88.8 (+0.4%) 16x slow	92.5	94.6 (+2.2%) 137x slow

Table 2.4: Parametric yield comparison for the sequential and exhaustive die matches with the previous work (Hungarian) and our algorithm (Ours). The clock skew margin is set to $0ps$.



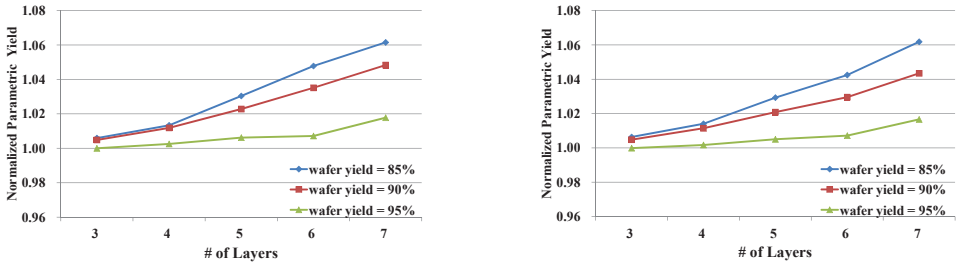
(a)



(b)

Figure 2.6: Normalized average mean and standard deviation of clock skew comparison for the sequential and exhaustive die matches with the previous work (Hungarian) and our algorithm (Ours).

exhaustive matching sequence: Fig. 2.6.(a) and (b) illustrate the comparison of normalized average mean and standard deviation of clock skew synthesized by the combinations of 2 matching sequences (e.g. sequential and exhaustive matching sequence) and 2-D die matching methods (e.g. ours and [3]) for 3-layered, 4-layered and 5-layered benchmark circuits, respectively. It is clearly revealed that the reduction of mean and std by any combinations of our proposed algorithm is much larger than the other combinations of matching sequences and 2-D die matching methods. Table 2.4 also shows the summary of the comparison of the yield improvement and run time increase when exhaustive matching sequence is explored in both ours and the algorithm in [3]. We can see that the yield improvement is relatively small by the exhaustive exploration of all matching sequences, but the run time increases significantly, which implies that attempting all possible matching sequence is not much effective in enhancing yield.



(a) skew margin : $0ps$

(b) skew margin : $5ps$

Figure 2.7: Normalized average parametric yield. The yields of individual wafer are set to 85%, 90% and 95%.

• **Assessing the quality of our subsequent-matching aware wafer matching algorithm**: Table. 2.7 and 2.8 show the comparisons of the yields and run times of IMH, ours, and ILP for 0ps and 5ps skew margin, respectively. Note that we are also not able to apply the ILP for more than 5-layered designs because the number of variables and

Benchmark	Method	3 Layer	4 Layer	5 Layer	6 Layer	7 Layer
	IMH	57.3	46.29	36.76	29.35	23.15
ispd09f11	Ours	57.62	46.83	37.92	30.74	24.72
	ILP	58.03	47.66	-	-	-
	IMH	57.24	46.2	36.83	29.32	23.45
ispd09f12	Ours	57.51	46.89	37.8	30.75	24.7
	ILP	57.93	47.51	-	-	-
	IMH	47.42	32.37	21.08	13.18	8.5
ispd09f21	Ours	47.55	32.77	22	14.32	9.04
	ILP	47.95	33.03	-	-	-
	IMH	38.44	22.65	12.49	6.65	3.41
ispd09f22	Ours	38.71	23.03	12.76	6.83	3.7
	ILP	38.87	23.22	-	-	-
	IMH	49.78	35.61	24.39	16.32	11.31
ispd09f31	Ours	50.02	35.94	25.46	17.61	11.94
	ILP	50.43	36.6	-	-	-
	IMH	56.4	45.06	35.41	27.78	21.87
ispd09f32	Ours	56.79	45.73	36.42	29.19	23.24
	ILP	57.16	46.37	-	-	-
	IMH	58.56	48.11	39.17	31.96	25.91
ispd09f33	Ours	58.97	48.8	40.28	33.28	27.49
	ILP	59.33	49.48	-	-	-
	IMH	53.75	41.51	31.35	23.69	17.67
ispd09f34	Ours	54.03	42.03	32.22	24.7	18.87
	ILP	54.6	42.81	-	-	-
	IMH	58.44	47.98	38.99	31.82	25.81
ispd09f35	Ours	58.85	48.62	40.13	33.15	27.38
	ILP	59.2	49.38	-	-	-
	IMH	58.62	48.16	39.23	32.04	25.97
ispd09fnb1	Ours	59.01	48.85	40.37	33.35	27.55
	ILP	59.38	49.55	-	-	-
	IMH	58.38	47.92	38.92	31.78	25.78
ispd09fnb2	Ours	58.77	48.56	40.07	33.1	27.33
	ILP	59.13	49.31	-	-	-

Table 2.5: Parametric yield of W2W matching achieved by IMH, our algorithm (Ours) and ILP. The clock skew margin and the yield of individual wafer are set to $0ps$ and 85%, respectively.

Benchmark	Method	3 Layer	4 Layer	5 Layer	6 Layer	7 Layer
ispd09f11	IMH	58.64	48.19	39.25	32.05	25.98
	Ours	59.03	48.88	40.39	33.37	27.56
	ILP	59.39	49.56	-	-	-
ispd09f12	IMH	58.64	48.19	39.25	32.05	25.98
	Ours	59.03	48.88	40.39	33.37	27.56
	ILP	59.39	49.56	-	-	-
ispd09f21	IMH	57.35	46.35	36.8	29.38	23.16
	Ours	57.66	46.88	37.95	30.79	24.75
	ILP	58.07	47.68	-	-	-
ispd09f22	IMH	56.02	44.36	34.4	26.65	20.61
	Ours	56.33	45.1	35.4	27.98	21.95
	ILP	56.71	45.67	-	-	-
ispd09f31	IMH	58.52	48.06	39.06	31.9	25.84
	Ours	58.91	48.69	40.21	33.22	27.43
	ILP	59.26	49.45	-	-	-
ispd09f32	IMH	58.64	48.19	39.25	32.05	25.98
	Ours	59.03	48.88	40.39	33.37	27.56
	ILP	59.39	49.56	-	-	-
ispd09f33	IMH	58.64	48.19	39.25	32.05	25.98
	Ours	59.03	48.88	40.39	33.37	27.56
	ILP	59.39	49.56	-	-	-
ispd09f34	IMH	58.64	48.19	39.25	32.05	25.98
	Ours	59.03	48.88	40.39	33.37	27.56
	ILP	59.39	49.56	-	-	-
ispd09f35	IMH	58.64	48.19	39.25	32.05	25.98
	Ours	59.03	48.88	40.39	33.37	27.56
	ILP	59.39	49.56	-	-	-
ispd09fnb1	IMH	58.64	48.19	39.25	32.05	25.98
	Ours	59.03	48.88	40.39	33.37	27.56
	ILP	59.39	49.56	-	-	-
ispd09fnb2	IMH	58.64	48.19	39.25	32.05	25.98
	Ours	59.03	48.88	40.39	33.37	27.56
	ILP	59.39	49.56	-	-	-

Table 2.6: Parametric yield of W2W matching achieved by IMH, our algorithm (Ours) and ILP. The clock skew margin and the yield of individual wafer are set to $5ps$ and 85%, respectively.

Method	metric	3 layers		4 layers		5 layers		6 layers		7 layers	
IMH	yield(%)	54.03	1.00	41.99	1.00	32.24	1.00	24.90	1.00	19.35	1.00
	run time(ms)	209.51	1.00	348.30	1.00	583.17	18.09	900.46	1.00	1306.15	1.00
Ours	yield(%)	54.35	1.01	42.55	1.01	33.22	1.03	26.09	1.05	20.54	1.06
	run time(ms)	2651.70	12.66	4560.30	13.09	7295.00	12.51	9886.40	10.98	13145.00	10.06
ILP	yield(%)	54.73	1.01	43.17	1.03	-	-	-	-	-	-
	run time(ms)	1807.90	8.63	554414.00	1591	-	-	-	-	-	-

Table 2.7: Average parametric yield comparison for the IMH, our algorithm (Ours) and ILP. The clock skew margin and the yield of individual wafer are set to $0ps$ and 85%, respectively.

Method	metric	3 layers		4 layers		5 layers		6 layers		7 layers	
IMH	yield(%)	58.28	1.00	47.67	1.00	38.57	1.00	31.30	1.00	25.22	1.00
	run time(ms)	223.49	1.00	382.74	1.00	598.53	1.00	913.52	1.00	1399.74	1.00
Ours	yield(%)	58.65	1.01	48.34	1.01	39.70	1.03	32.63	1.04	26.78	1.06
	run time(ms)	2817.20	12.61	4794.70	12.53	7409.80	12.38	10142.00	11.10	13455.00	9.61
ILP	yield(%)	59.01	1.01	49.02	1.03	-	-	-	-	-	-
	run time(ms)	1871.77	8.38	570494.00	1490	-	-	-	-	-	-

Table 2.8: Average parametric yield comparison for the IMH, our algorithm (Ours) and ILP. The clock skew margin and the yield of individual wafer are set to $5ps$ and 85%, respectively.

constraints exceeds the range that the ILP solver can handle. It is clearly revealed that our wafer-matching algorithm is able to improve the yield by 0∼8% over the previous work (IMH) for 3∼7-layered designs while spending a little run time in the experimental result. In Fig. 2.7, we investigate the impact of the yield of individual wafer to the yield improvement of our algorithm over the previous method. The results show that our algorithm can achieve the more yield improvement as the layers of designs are increased while the individual wafer yield is decreased.

2.6 Summary

In this work, we presented comprehensive die-to-die/wafer-to-wafer matching algorithms to improve the parametric yield of 3D ICs. The key part is to take into account the effect of current two-dimensional die/wafer matching on the result of subsequent two-dimensional die/wafer matching sequence, so that a globally maximal multi-dimensional matching can be achieved. Specifically, we formulated the two-dimensional die/wafer matching problem into a problem of finding a maximum flow of minimum cost in a network and solved it optimally in polynomial time. From the experimental results using benchmark circuits, it was shown that the proposed algorithm was able to improve the parametric yield by 2%, 7%, and 8% for 3-layered, 4-layered, and 5-layered 3D ICs, respectively. Finally, the following two issues should be addressed in the future: (1) since as the number of layers to be stacked increases to 8 or beyond, the order of two-dimensional die/wafer matching sequence becomes important, it is needed to address the problem of finding a good order in this prospective; (2) since the body biasing applied to dies may change the power consumption, a more delicate die-to-die matching is required to control the power consumption.

Chapter 3

Edge Layer Embedding Algorithm for Mitigating On-Package Variation in 3D Clock Tree Synthesis

In this chapter, we analyze the effect of on-package variation on 3D clock trees and address the problem of on-package variation aware layer embedding in 3D clock tree synthesis. We show hat a careful clock edge embedding can greatly reduce the impact of on-package variation on the 3D clock skew, thereby enhancing chip yield, and propose a two-step solution for the problem of on-package variation aware layer embedding of clock edges.

3.1 Introduction

As CMOS process technology scales down, the control of process variation induced delay variability becomes highly important not to worsen the chip yield. In 3D ICs, the die-to-die (D2D) as well as within-die (WID) variation can cause on-package (within-

chip) variation when the process of stacking multiple dies, which might exhibit different process corners, is performed to form 3D IC package. In 3D ICs, the variability of the difference between the delays of two clock signal paths routed on different dies each other can be enlarged, whereas the variability of the delay of a logic signal path across multiple dies can be reduced due to an averaging effect. Thus, integrating the dies without considering on-package variation can cause clock skew violation and lead to a low chip yield [24]. Recently, Kim and Kim [3] proposed a die-to-die matching algorithm with the consideration of body biasing as a post-silicon tuning to enhance the parametric yield. However, there exists an achievable range that the die matching algorithm can mitigate the on-package variation because its concern is only the elimination of mismatches of dies having different process corners. On the other hand, Chae and Mukhopadhyay [17] proposed a level shifter insertion method with the tier adaptive voltage scaling as post-silicon tuning. They attempted to explore all the discrete insertion candidates of each level shifter in 3D paths and placed the shifter to the location that was likely to enable to minimize the delay variation of 3D paths. However, the problem of finding exact location at which the level shifters are placed to minimize the delay variation of critical path in a circuit has not been fully investigated. Recently, Kim, Joo, and Kim [18] proposed adjustable delay buffer (ADB) insertion method to reduce clock skew variation in multiple supply voltage designs. Even though the use of ADBs is effective in reducing clock skew variation, the work is confined to 2D IC designs.

It should be noted that in the area of 2D clock tree synthesis, an extensive effort has been made to mitigate the clock skew variation (e.g., [21–23]). A common feature among many of the variation-aware 2D clock tree synthesis methods is that they try to find proper (x, y) location of clock tree nodes by integrating Statistical Static Timing Analysis (SSTA) into Deferred-Merge Embedding (DME) algorithm [19]. Recently, in the 3D clock tree synthesis flow, a number of works have observed that the layer

assignment of clock tree edges would help reduce the skew variability [6, 14]. Hu *et al.* [14] demonstrated empirically under the use of 3D H-tree structure that the skew variation between a pair of clock sinks can be reduced if the sinks and their connecting edges are all together placed in the same layer. Zhao *et al.* [6] further elaborated the idea in that they aggressively attempted to route the most part of clock paths to the same layer at the expense of the increase of TSV usage. In addition, Yang *et al.* [15] proposed an on-package variation aware 3D clock tree synthesis methodology to reduce the clock signal variation. The idea is to place the clock buffers in clock tree on as many uncorrelated dies as possible during the process of clock tree synthesis. Even though this method is effective in mitigating the effect of on-package variation on the critical path, it demands much more TSVs than necessary. In addition to the variation aware yield enhancement techniques, a number of 3D clock tree synthesis methodologies have addressed another yield problems [30–32]. For example, to support pre-bond testability, 3D clock tree synthesis with pre-bond testability is introduced by [30] and optimized by [31]. Furthermore, Liu *et al.* [32] propose a TSV-aware 3D clock tree synthesis methodology that takes into account the density constraint of TSVs, and their parasitic and coupling effects.

In this paper, we propose an effective method to solve the new problem of on-package variation aware clock edge embedding with the objective of minimizing the clock skew variation while taking into account both the usage of TSVs and the effect of on-package variation, which has never been addressed in the prior works even though this topic is not only unique to the 3D IC design but also highly important. The contributions of our work are summarized as:

- We address a new problem of edge embedding to layers in 3D clock tree synthesis to mitigate the effect of on-package variation on the clock skew, and propose a graph-based algorithm (called *full embedding*) that maximizes the layer sharing among the edges of clock paths followed by a local refinement (called

partial embedding) to further reduce the clock skew.

- A full 3D clock tree synthesis flow from the clock topology generation down to the clock routing and buffering that makes use of the on-package variation aware edge embedding is developed together with a possible integration of the node embedding into the proposed edge embedding.

- Extensive experiments are conducted to show how much the edge embedding to layers is effective and valuable in reducing the effect of on-package variation, so that it can help the clock skew be controlled within the yield constraint.

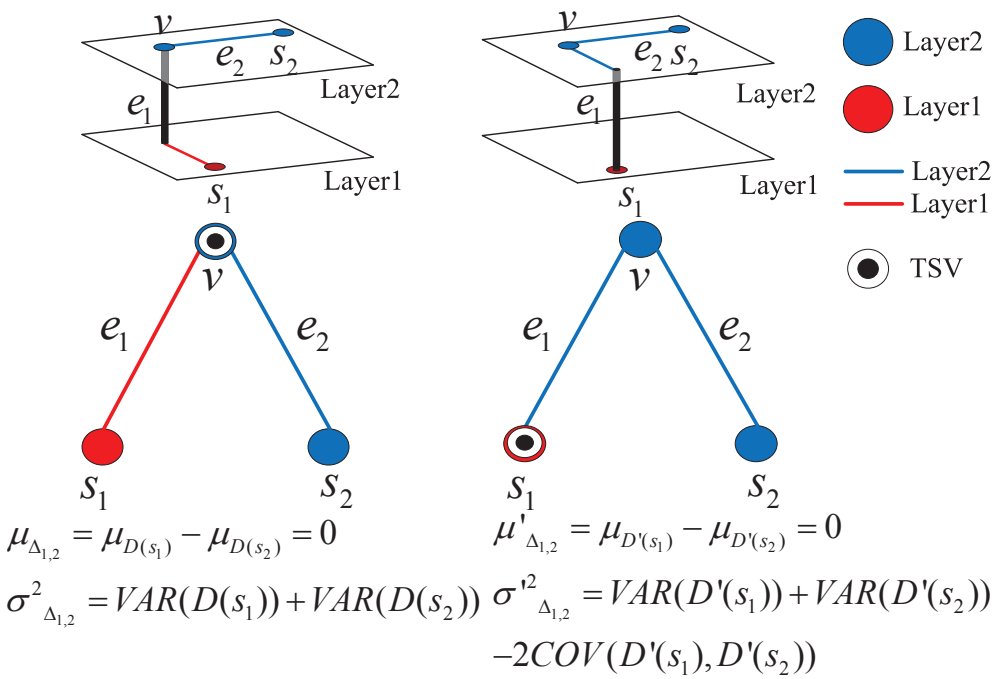
The remainder of this paper is organized as follows. The 3D clock edge embedding problem for minimizing skew variation and the motivation of the work are given in section 3.2. In section 3.3, our proposed on-package variation aware edge embedding algorithm that is performed in two steps is presented. Section 3.4 provides a set of experimental data to assess the effectiveness of the proposed method. Finally, a conclusion of our work is given in section 3.5.

3.2 Problem Definitions and Motivation

The 3D clock edge embedding problem for minimizing skew variation can be described as:

Problem 1. Clock edge embedding problem: *For an input 3D clock tree in which the clock node embedding process has been done, assign the clock edges to layers so that the clock skew variation is minimized while maintaining the TSVs defined by the node embedding.*

First, we illustrate how the clock skew variation in a 3D clock tree can be changed as the clock edges are embedded on the layers.



(a) Without layer sharing

(b) With layer sharing

Figure 3.1: Clock skew comparison between two different edge embeddings. (a) An edge embedding without layer sharing between edges e_1 and e_2 under TSV constraint of 1. (b) An edge embedding with layer sharing between edges e_1 and e_2 under TSV constraint of 1.

Let us consider two sinks (e.g., flip-flops) s_i and s_j that are driven by the clock signal on a clock tree. Let $D(s_i)$ and $D(s_j)$ denote the arrival times of clock signal at s_i and s_j, respectively, and assume that the arrival time difference, $\Delta_{i,j}$, between $D(s_i)$ and $D(s_j)$ follows the Gaussian distribution:

$$\Delta_{i,j} = D(s_i) - D(s_j) \sim N(\mu_{\Delta_{i,j}}, \sigma_{\Delta_{i,j}}), \tag{3.1}$$
$$\mu_{\Delta_{i,j}} = \mu_{D(s_i)} - \mu_{D(s_j)},$$
$$\sigma^2_{\Delta_{i,j}} = VAR(D(s_i)) + VAR(D(s_j)) - 2COV(D(s_i), D(s_j)).$$

Then, according to the expression in *Eq.1*, we can derive the distribution of clock skew $\Delta_{1,2}$ for two different clock edge embeddings: embedding clock edge e_1 to layer 1 and embedding e_1 to layer 2. We can observe that since the correlation between random variables $D(s_1)$ (defined by e_1) and $D(s_2)$ (defined by e_2) come from the variation in technology parameters, which will be uncorrelated, if e_1 and e_2 are placed on different layers as shown in Fig. 3.1(a), $D(s_1)$ and $D(s_2)$ are independent, i.e., $COV(D(s_1), D(s_2)) = 0$ [14]. On the other hand, the random variables $D(s_1)$ and $D(s_2)$ are correlated if edges e_1 and e_2 are both placed on the same layer as shown in Fig. 3.1(b). Precisely, since the within-die variation is spatially correlated, the variance of skew distribution $\sigma'_{\Delta_{1,2}}$ in Fig. 3.1(b) would be much smaller than $\sigma_{\Delta_{1,2}}$ in Fig. 3.1(a)[1]. We observe from the analysis that *by embedding the edges on two clock paths into the common layers as many as possible, the resulting clock skew variation between the two paths can be reduced.*

As motivated by the example in Fig. 3.1, we solve the clock edge embedding problem of mitigating on-package variation in two steps. In the first step, we maximize the total amount of layer sharing among the edges on all clock signal paths from the clock source to sinks. This leads to *minimize* the impact of on-package variation on *the total sum of the variations* of delay difference between all clock paths. In the second

[1]Note that $COV(D'(s_1), D'(s_2)) > 0$.

step, the edge embedding result obtained in the first step is refined locally in order to *reduce the worst variation* of the clock skew. It should be noted that this work focuses on-package variation aware edge embedding only without integrating variation aware buffer insertion. However, our experimental results shown in section 3.4 are all the ones produced by the application of the conventional buffer insertion afterwards to precisely evaluate how much our edge embedding algorithm by itself is effective.

Before describing our proposed two-step algorithm in the next section, we introduce notations and terminologies to be used in the presentation. We define P_i as the path from $root$ to a sink s_i in a clock tree $T(S)$ where S is the set of sinks. P_i can be expressed as a collection of edges, $\bigcup_{e_j \in P_i} e_j$. Let E denote the set of edges on $T(S)$. Then, we can define $CP_{i,j}$ to be the amount of common layer sharing between P_i and P_j. To formulate $CP_{i,j}$, we introduce an operator $< e_i, e_j >$ as

$$< e_i, e_j >= \begin{cases} 1 & \text{if } l(e_i) = l(e_j), \\ 0 & \text{otherwise.} \end{cases} \tag{3.2}$$

$CP_{i,j}$ is then expressed as:

$$CP_{i,j} = \sum_{m \in P_i} \sum_{n \in P_j} < e_m, e_n > . \tag{3.3}$$

Then, we define the total amount of layer sharing TLS as:

$$TLS = \sum_{\forall (s_i, sj):\ s_i, s_j \in S} CP_{i,j}. \tag{3.4}$$

Since some edges appear more than one clock path, TLS can be re-expressed as:

$$TLS = \sum_{\forall (e_i, e_j):\ e_i, e_j \in E} w_{i,j} < e_i, e_j > . \tag{3.5}$$

where $w_{i,j}$ is the number of pairs of clock paths where e_i is in one path and e_j is in the other path.

3.3 The Proposed Algorithm for On-Package Variation Aware Edge Embedding

Fig. 3.2 shows a comparison of the conventional 3D clock tree synthesis flow without considering variation effect during edge embedding (e.g., [5]) and our flow that takes into account the on-package variation. After the location and length of TSVs to be allocated are determined by the outcome of the clock node embedding, the conventional flow assigns layers to edges based on the easiness of routability and TSV placement. On the other hand, our proposed flow includes one more consideration to mitigate on-package variation during the edge embedding stage According to the designer's decision, some of the edges are frozen to specific layers while the rest of edges are applicable to the on-package variation aware embedding process. In the following subsections, we describe our three-step approach called LEE-CLK (Layer Embedding of Edges in 3D CLocK tree): (1) maximizing the sharing of layers among the edges, (2) refining the layer assignment of edges to further reduce the worst latency difference (i.e., clock skew) and (3) clock routing and buffer insertion.

3.3.1 Algorithm for Maximizing Layer Sharing of Edges

The problem we want to solve in the first step can be described as:

Problem 2. Maximum layer sharing (MLS) of edges: *For an input 3D clock tree in which the clock node embedding process has been done, assign the clock edges to layers so that the total amount of layer sharing by edges (i.e., the quantity of TLS in Eq.3.5) is maximized while maintaining the TSVs defined by the node embedding.*

From the result of node embedding, we can extract the layer candidates on which each edge can be placed. We call an embedding of an edge between two nodes in a clock tree to a layer *a full embedding* if the edge is entirely embedded on the layer, and call it *a partial embedding* if the edge is partially embedded on the layer. In this step

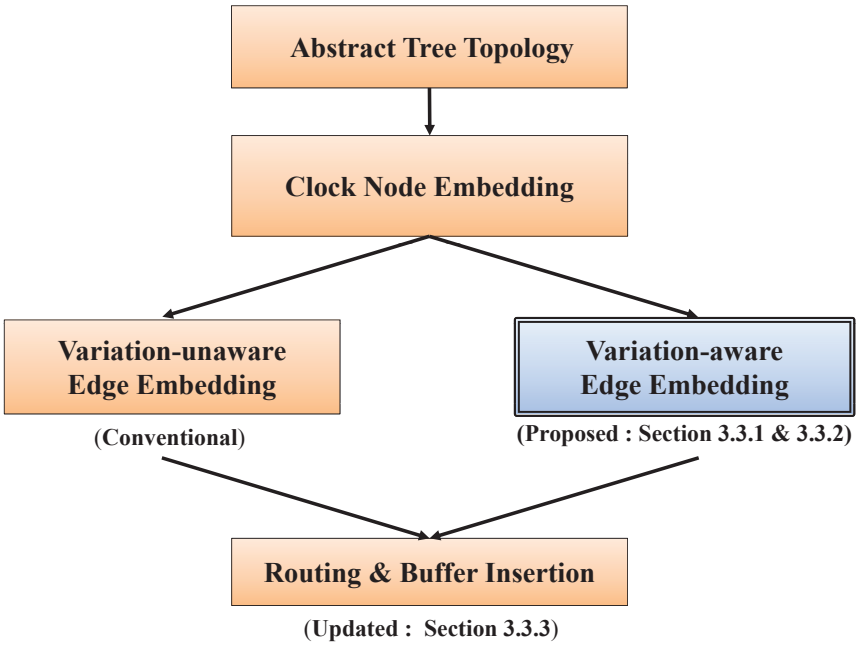


Figure 3.2: 3D clock tree synthesis flow comparison between the conventional flow and our proposed flow.

of our algorithm considers full embeddings only, while in the second step it refines the results by utilizing partial embeddings. Fig. 3.3 shows an example that illustrates how the layer candidates of an edge are identified. Consider an internal clock node a and two child nodes b and c of a that are placed and positioned in layers by the process of node embedding. When a is placed in between the layers in which b and c are placed as shown in Fig. 3.3, the range of layer candidates for embedding edge $e_{a,b}$ is $\{l_j, \cdots, l_k\}$ and the range for $e_{a,c}$ is $\{l_i, \cdots, l_j\}$.

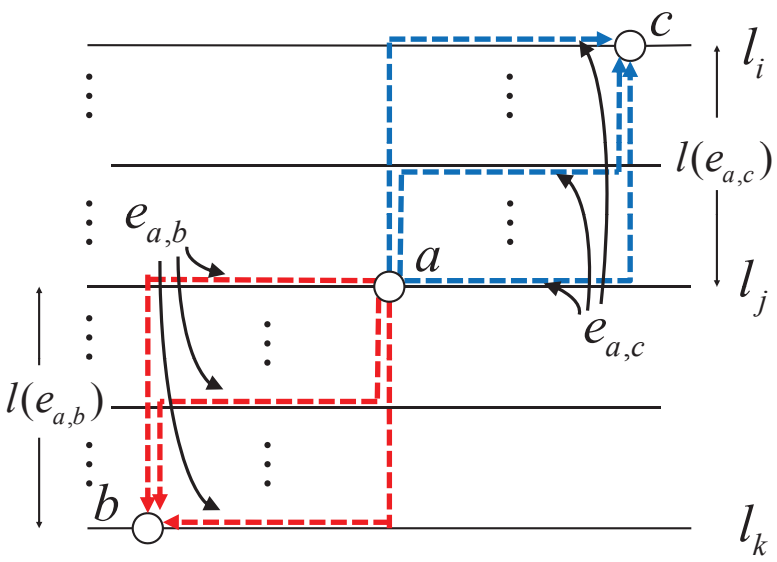


Figure 3.3: The identification of embedding ranges of layers for edge $e_{a,b}$ and $e_{a,c}$, in which nodes b and c are the two children of a clock node a.

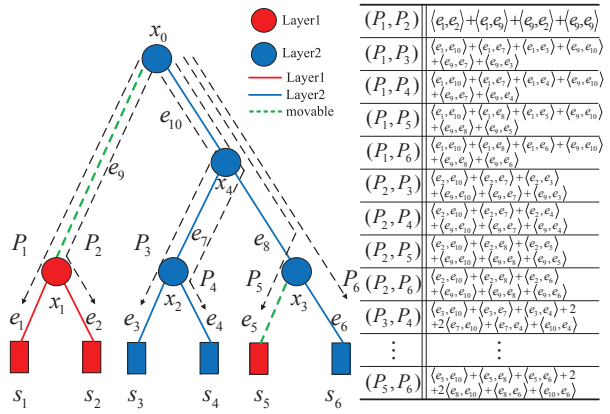
We use notation $range(e_i)$ of an edge e_i in the clock tree to indicate the set of layers on which e_i can be embedded and $|range(e_i)|$ to indicate the number of the layers in the range. For example, the left side in Fig. 3.4(a) shows the 2-layered clock tree after the process of node embedding. The clock tree has six sinks s_1, s_2, s_3, s_4, s_5 and s_6, in which s_1, s_2 and s_5 are placed in layer l_1 and s_3, s_4 and s_6 are in layer l_2.

We can observe that by the layer embedding of the internal nodes x_1, x_2, x_3 and x_4 in the clock tree, two TSVs, one from x_0 to x_1 and the other from x_3 to s_5, are allocated. Consequently, under the TSV allocation, edge e_9 that is on path P_1 (= $e_9 \rightarrow e_1$) and P_2 (= $e_9 \rightarrow e_2$) has $range(e_9) = \{l_1, l_2\}$ and $|range(e_9)| = 2$. Likewise, edge e_5 that is on path P_5 (= $e_{10} \rightarrow e_8 \rightarrow e_5$) has $range(e_5) = \{l_1, l_2\}$. Edges e_1 and e_2 both have $range(.) = \{l_1\}$ and $|range(.)| = 1$. The rest of edges have $range(.) = \{l_2\}$ and $|range(.)| = 1$.

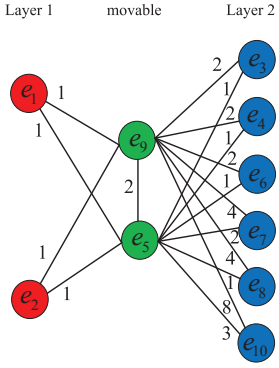
We formulate the problem of maximal layer sharing of edges into a multi-way graph partitioning problem in a graph $G(V, E, W)$, which we call the *layer-compatible graph* of the input clock tree. The construction of $G(V, E, W)$ is performed by the following steps:

1. (**Construction of common edge relation graph**) We generate a graph $G_0(V', E', W')$ where each node in V' represents a distinct edge in the clock tree and there is an edge in E' between nodes v_i and v_j if $range(e_i) \cap range(e_j) \neq \phi$ where e_i and e_j are the edges in the clock tree corresponding to nodes v_i and v_j, respectively. We assign weight $w_{i,j} \in W'$ on edge $(v_i, v_j) \in E'$ such that $w_{i,j}$ is the number of occurrences of $< e_i, e_j >$ pairs in all possible combinations of two paths among the clock signal paths from the clock source to sinks. For example, the table in Fig. 3.4.(a) shows the path combinations together with their edge pairs. Then, $w_{7,9} = 4$ since $< e_7, e_9 >$ occurs in path combinations (P_1, P_3), (P_1, P_4), (P_2, P_3) and (P_2, P_4), as shown in the table. Fig. 3.4(b) shows the graph $G_0(V', E, W')$ constructed from the input clock tree in Fig. 3.4(a).

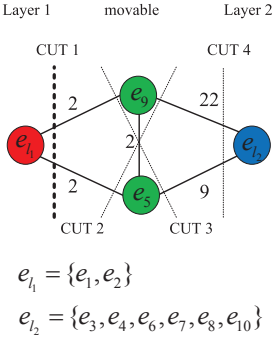
2. (**Derivation of layer-compatible graph**) We group the nodes in V' whose connecting edges in clock tree have $|range(.)| = 1$ and have already been assigned to the same layer. We replace the grouped nodes with a new super node. Note that the range of a super node is identical to that of the nodes before grouping.



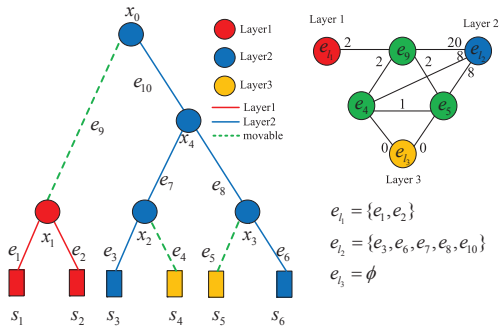
(a) Layer candidates of edge embedding



(b) Common edge relation graph G_0 from (a)



(c) Layer-compatible graph G from (b) and 2-way min-cut



(d) Another layer-compatible graph with a dummy node e_{l_3}

Figure 3.4: Formulation of the maximal layer sharing problem into a graph partitioning problem. (a) The input clock tree after the layer embedding of nodes and the clock edge pairs for layer sharing. (b) The construction of an initial graph derived from the input clock tree in (a). (c) The final graph refined from the graph in (b) by adding or updating nodes, edges, and weights. (d) Another input clock tree and layer-compatible graph with a dummy node.

The edge weight between a super node and a (outside) node v_l is set to the sum of the edge weights between the nodes before grouping and the replaced super node. Then, the layer-compatible graph $G(V, E, W)$ is the one obtained from $G_0(V', E', W')$ by the process performed in this step. For example, Fig. 3.4(c) shows the graph $G(V, E, W)$ of $G_0(V', E', W')$ in Fig. 3.4(b), in which the six nodes with blue color are abstracted into a super node and the weights of the connected edges to the super node are updated accordingly.

3. (**Inclusion of dummy nodes and graph partitioning**) We check the nodes in V whose connecting edges have $|range(.)| = 1$, and collect the layers to which the edges are assigned. Let R be the set of layers collected and reset $\bar{R} = \{l_1, l_2, \cdots, l_k\} - R$. Then, for each layer $l_k \in \bar{R}$, a dummy node with its $range(.) = \{l_k\}$ is added to V, and an edge from the dummy node to every node whose range overlaps with that of the dummy node, the edge weight is set to 0. This procedure ensures that for each layer l_k, the updated graph G has exactly one super node or one dummy node (but not both) with $range(.) = \{l_k\}$. The graph on the right side in Fig. 3.4(d) shows an example of layer-compatible graph with a dummy node e_{l_3} that is derived from the clock tree on the left side in Fig. 3.4(d).

Once the graph $G(V, E, W)$ is obtained, we partition the graphs G into K parts for a K-layered clock tree such that (*constraint-1*) each part should contain exactly one dummy node or one super nodes (but not both) and (*objective-1*) the sum of the edges on the cut should be *minimized*. The *constraint-1* ensures the correctness of the layer embedding of edges and the *objective-1* ensures the *maximization* of the layer sharing by edges. For example, the cuts in Fig. 3.4(c) represent all possible ways to partition the graph. For example, cut-2 indicates the edge layer embedding that assigns e_9 to layer 1 and e_5 to layer 2. Among the cuts in Fig. 3.4(c) cut-1 returns the maximal layer

sharing among the edges, i.e., the maximum of TLS in *Eq.3.5*, which is 2 + 22 + 9 + w_f = 33 + 47 = 80 where w_f denotes the sum of weights between *fixed* edges in the clock tree.

Since our graph partitioning problem does not have to care about balanced partitioning for $K = 2$, the problem is reduced to an ordinary *s-t min-cut* problem, which can be solved optimally in polynomial-time [33]. However, for $K > 2$, the problem is intractable. Note that there are numerous algorithms that have addressed the K-way partitioning problem [34] in the literatures. In this work, we use a variant of the well-known Kernighan-Lin (K-L) partitioning algorithm [35] in which we extend the K-L algorithm to include seeds (i.e., dummy nodes and super nodes) in the initial partition and to allow the multi-way moves of each node. The generation of layer-compatible graph can be done in $O(|E|^2)$ where the dominant part is the computation of weights in W and $|E|$ is the number of edges in the clock tree, and the time complexity of the K-L based graph partitioning algorithm is bounded by $O(|E| \cdot K)$.

Embedding edges combined with node embedding: Depending on the node embedding algorithm adopted, the integration of edge embedding into the node embedding would be quite different. Here, we propose a solution when DLE-3D (Deferred-Layer Embedding 3D) algorithm in [5], which ensures the TSV allocation of minimal cost, is used as the node embedding. Since DLE-3D collects the set of feasible layers on which the root of the clock tree can be placed while still maintaining the minimal use of TSVs, we can apply our edge embedding algorithm to every clock tree that is generated by DLE-3D, varying the layer assignment of root. Then, we choose the clock tree with node and edge embedding that has the largest TLS value. (The edge embedding algorithm combined with the node embedding algorithm [5] is summarized in **Algorithm 1**.)

Algorithm 2 Integration of our edge embedding with the node embedding algorithm in [5].

1: *Input*: a 3D clock tree topology $T(V_T, E_T)$;
2: *Output:* an embedding of nodes (V_T) and edges (E_T) in T;
3: • $[l^L_{root}, l^U_{root}] \leftarrow$ *Apply the bottom-up traversal* in DLE-3D [5]; // feasible range of layers
4: • $min\text{-}cut \leftarrow \infty$; // minimizing cut means maximizing sharing of layers
5: **for** $l_{root} \leftarrow$ from l^L_{root} to l^U_{root} **do**
6: • $l^*(V_T) \leftarrow$ *Apply the top-down traversal* in DLE-3D [5]; // embedding of nodes
7: • *Construct the layer-compabile graph* $G(V, E, W)$ from T with $l^*(V_T)$; // the three steps in Sec. 3.3.1
8: • $l^*(E_T) \leftarrow$ *Apply K-way min-cut* to $G(V, E, W)$;
9: **if** $(cut - size(l^*(E_T)) < min\text{-}cut)$ **then**
10: • $min\text{-}cut \leftarrow cut\_size(l^*(E_T))$;
11: • $l(V_T, E_T) \leftarrow l^*(E_T) \cup l^*(V_T)$;
12: **end if**
13: **end for**
14: **return** $l(V_T, E_T)$;

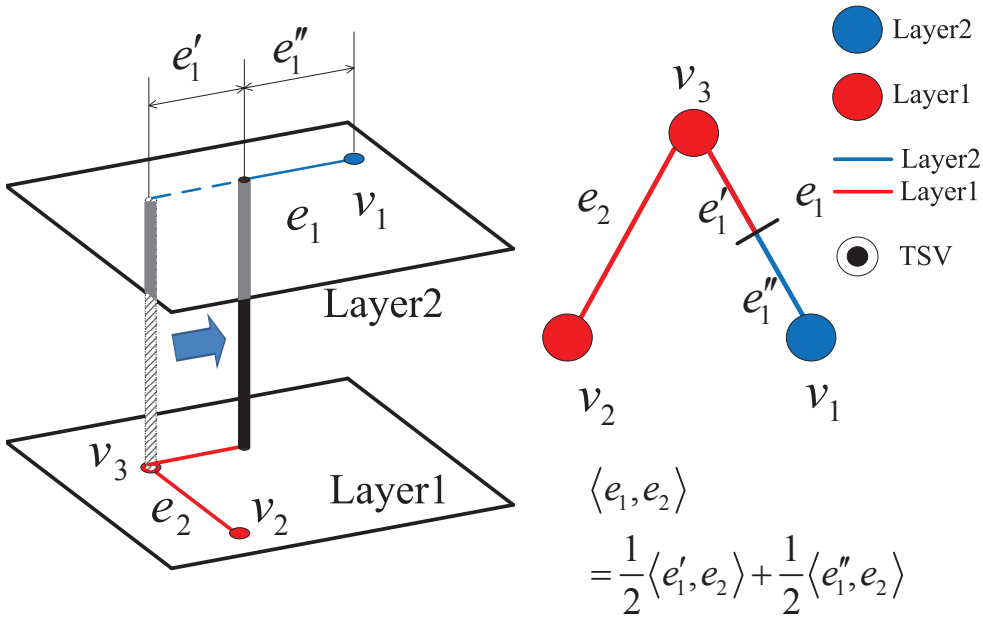
3.3.2 Refinement: Partial Edge Embedding on Layers

The maximization of layer sharing by the edges in the clock tree performed by the algorithm in section 3.3.1 will lead to reduce the effect of the on-package variation caused by the multiple stacked dies on the delay difference of *most* pairs of the clock paths from the clock source to the sinks. However, it does not mean that the largest delay difference (i.e., clock skew) is always minimal. The clock skew constraint may not still be met due to the on-package variation. In this refinement step, we want to reduce the clock skew, if there is a skew violation, by trading a part of the layer sharing with a part of clock skew.

Our refinement algorithm of edge embedding is an iterative one, in which re-embedding of edges is repeatedly tried to incrementally reduce the clock skew at the expense of the decrease of the total sharing of layers. Our algorithm introduces a parameter, called *embedding granularity* ρ, which is initially set to 1, i.e., at the first iteration, 1/2 at the second iteration, 1/4 at the third iteration, $\cdots$, until the clock skew constraint is satisfied or there is no further reduction of the clock skew. For example, when $\rho = 1/2$ at the second iteration, the edge e_1 in Fig. 3.5(b) is split into two parts e_1' and e_1'' of equal length, and the parts are attempted to be assigned to different layers, as shown in Fig. 3.5(a). Thus, as the iteration progresses, our refinement algorithm performs a more fine-grained edge re-embedding. (See Fig. 3.6 for illustration.) Our refinement works as follows:

1. $\rho = 1$.
2. Extract the two clock paths that generate the largest layer difference[2] CP^L. Then, for each segment of the edges in the paths where the segment size is

[2]To be more accurate, the two clock paths should be selected based on the statistical analysis of the worst clock skew, but, at this point, it is very likely that choosing the two clock paths with the largest layer difference corresponds to that clock skew.



(a) Partial embedding on two layers

(b) Calculation of $< e_1, e_2 >$ for (a)

Figure 3.5: An example showing layer embeddings using embedding granularity $\rho = 1/2$. (a) Embedding of the two segments of edge e_1 on two layers. (b) The computation of layer sharing operator $< e_1, e_2 >$ (in *Eq.2*) for $\rho = 1/2$.

defined by the ρ value, all possible layer re-embeddings are tried and the cost, δ, of each re-embedding is computed:

$$\delta = \frac{TLS_{before} - TLS_{after}}{CP^L_{before} - CP^L_{after}}. \tag{3.6}$$

where TLS_{before} and TLS_{after} represent the amount of total sharing of layers (i.e, *Eq.3.5*) and CP^L_{before} and CP^L_{after} represent the largest layer difference before and after the corresponding re-embedding of edge segment, respectively.[3] Then, among the re-embeddings of edge segments, we choose a re-embedding with the smallest value of δ, and perform the re-embedding.

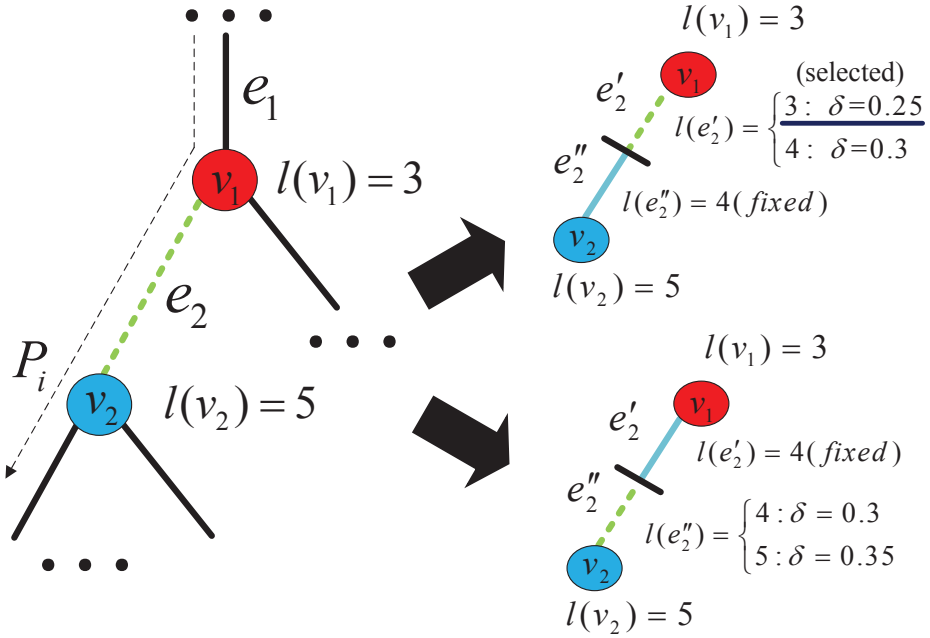
3. If a re-embedding is done in step 2 and it results in meeting the CP^L constraint, stop successfully; if no re-embedding is performed in step 2, report that the CP^L cannot be reduced any more and stop; otherwise, set $\rho = \rho/2$ and go to step 2.

The time complexity of each iteration is bounded by $O(|E|(1/\rho)K(|E|(1/\rho) + 2|E|)$ in which $|E|(1/\rho)+2|E|$ accounts for the computation of $< e_i, e_j >$ operations and $|E|(1/\rho)K$ accounts for the number of re-embeddings tried. However, many edges are fixed or have a few layers to re-embed, the complexity is significantly reduced in practice.

3.3.3 Clock Tree Routing and Buffer Insertion

In this step, we extend the DME-3D algorithm proposed in [5] to find the (x, y) location of nodes and edges in 3D abstract clock tree without violating the maximum loading capacitance and clock skew constraints. The difference between our approach and DME-3D is that the edges of our approach is partially divided in a ratio of ρ_V while the edges of DME-3D are fully embedded to the layer to which their parent node is

[3]Note that the δ value is computed only when $CP^L_{after} < CL^L_{before}$ because the inequality means the clock skew is unlikely to be reduced.



(a) e_2 has three candidates (b) Partial embedding of e_2 in (a) and calculation of δ

Figure 3.6: An example illustrating the partial layer embedding when $\rho = 1/2$. (a) Edge e_2 has three layer candidates, $\{l_3, l_4, l_5\}$. (b) The partial layer embedding for $l(e'_2)$ is chosen for the smallest δ value.

assigned. Fig. 3.7 describes 1-D view of 3D clock routing from the roots of two subtrees rooted at a and b to a merging point v. Since we newly include the edge layer assignment and embedding granularity ρ to the 3D abstract clock tree, we modify the calculation of the merging point. For $0 \leq x \leq L = d(ms(a), ms(b))$ where x is the possible loci where v can be placed, we drive x in *Eq.3.9* by equating the delay in *Eq.3.7* from v to the sinks in the subtree rooted at a to the delay in *Eq.3.8* from v to the sinks in the subtree rooted at b. If $x < 0$ (or $x > L$), we set $|e_a| = 0$ (or $|e_a| = L'$) and $|e_b| = L'$ (or $|e_b| = 0$) where L' is computed by *Eq.3.10* (or *Eq.3.11*).

$$t(v, T_a) = t_a + \frac{1}{2} r_w \cdot c_w \cdot x^2 + \alpha_1 \cdot x + \alpha_2, \tag{3.7}$$

$$\alpha_1 = r_v \cdot c_w \cdot (l_v - l_a) + (r_w \cdot c_v - r_v \cdot c_w)(l_{e_a} - l_a) \cdot \rho_a + r_w \cdot C_a,$$
$$\alpha_2 = \frac{1}{2} \cdot r_v \cdot c_v \cdot (l_v - l_a)^2 + r_v \cdot C_a \cdot (l_v - l_a).$$

$$t(v, T_b) = t_b + \frac{1}{2} r_w \cdot c_w \cdot (L - x)^2 + \beta_1 \cdot (L - x) + \beta_2, \tag{3.8}$$

$$\beta_1 = r_v \cdot c_w \cdot (l_b - l_v) + (r_w \cdot c_v - r_v \cdot c_w)(l_b - l_{e_b}) \cdot \rho_b + r_w \cdot C_b,$$
$$\beta_2 = \frac{1}{2} \cdot r_v \cdot c_v \cdot (l_b - l_v)^2 + r_v \cdot C_b \cdot (l_b - l_v).$$

$$x = \frac{t_b - t_a + \beta_2 - \alpha_2 + \frac{1}{2} r_w \cdot c_w \cdot L^2 + \beta_1 \cdot L}{\alpha_1 + \beta_1 + r_w \cdot c_w \cdot L}. \tag{3.9}$$

$$|e_a| = L' = \frac{-\alpha_1 + \sqrt{\alpha_1^2 - 2 r_w \cdot c_w \cdot (\alpha_2 - \beta_2 + t_a - t_b)}}{r_w \cdot c_w}. \tag{3.10}$$

$$|e_b| = L' = \frac{-\beta_1 + \sqrt{\beta_1^2 - 2 r_w \cdot c_w \cdot (\beta_2 - \alpha_2 + t_b - t_a)}}{r_w \cdot c_w}. \tag{3.11}$$

In addition to the edge embedding, since the process variation of clock buffers is another dominant factor for clock skew variations, we try to minimize buffer insertions

using the greedy method in [36], in which buffers are inserted when the loading capacitance exceeds the maximum loading capacitance constraint or when a certain amount of detouring wire reduction is expected.

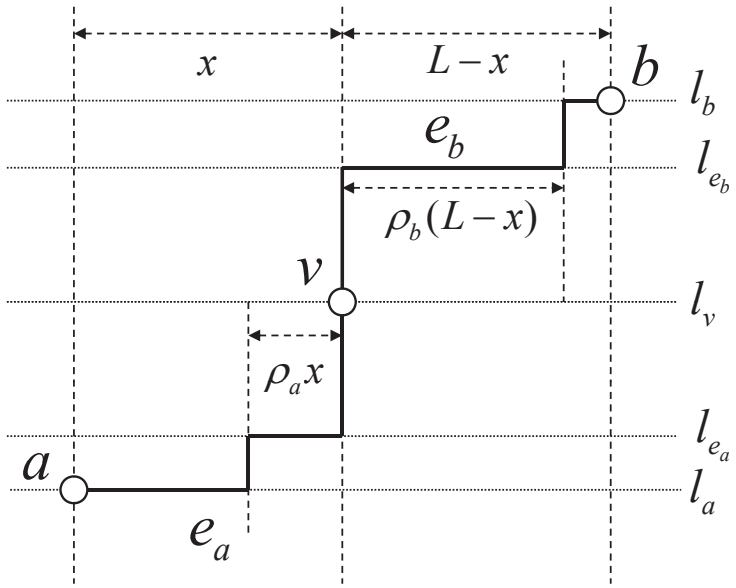


Figure 3.7: 1-D view of 3D clock routing from internal nodes a and b to a merging point v with partial embedding parameter ρ_a and ρ_b for e_a and e_b, respectively.

3.4 Experimental Results

The proposed algorithm LEE-CLK (layer embedding of edges in 3D clock tree) has been implemented using C++ and run on Linux machine with 2.66 GHz Intel Core2 Duo Processor and 2GB memory. We evaluate the effectiveness of our algorithm using the benchmark circuits in ISPD 2009 clock network synthesis contest [28] under Elmore delay model. Since the benchmark circuits are originally designed for 2D clock

tree synthesis, we transformed the benchmark circuits into 2-layered and 4-layered 3D circuits with the reduction of die length by a factor of $\sqrt{2}$ and 1/2, respectively. The layers of sinks in the benchmark circuits are randomly assigned. Technology parameters are based on 45nm predictive technology [29]. We perform 1000 Monte Carlo runs in HSPICE simulation with random parameters for the variations in the channel length of buffers, the wire width, and the sink input capacitance. (The random variables are modeled according to that in [6].) Each random variable consists of the nominal value, D2D variation, and WID variation. A variable x_i in die-i follows

$$G_i \sim N(0, \sigma_{D2D}^2), \tag{3.12}$$

$$x_i \sim G_i + N(0, \sigma_{WID}^2) + x_i^0 \tag{3.13}$$

where σ_{D2D}, σ_{WID}, and x_i^0 are D2D variation, WID variation, nominal value of x_i, respectively, and σ_{D2D} and σ_{WID} are set to 5% of the nominal value. The spatial correlation of WID variation is modeled using quadratic tree partitioning as does in [37]. We used a quadratic tree of 5 levels which divides a layer into 32X32 grids for spatially correlated WID with level-0 for uncorrelated WID. To evaluate the effect of the reduction of skew variation on the chip yield, we define a parametric yield $Y = \sum_{i=1}^{N} x_i/N$ where $x_i = 1$ if $|skew_{max}| \leq B_{skew}$ and 0, otherwise, in which B_{skew}, $skew_{max}$ and N represent the skew bound, the global skew, and the size of Monte Carlo runs, respectively.

• **Assessing the quality of the proposed algorithm**: Tables 3.1 and 3.2 summarize the comparisons of the number of TSVs, the number of buffers and the latency for 2-layered and 4-layered designs, respectively. Note that LEE-CLK uses slightly longer clock latency (2%) over that by [5]. This is because every TSV allocated by [5] is always positioned at the beginning of the driving edge connected to the TSV to minimize the delay of clock signal, but our algorithm explores many alternative positions. Tables 3.3,3.5, 3.7, and 3.9 show that our algorithm reduces the mean and standard de-

Table 3.1: Comparison of results by [5, 6] and LEE-CLK with (w/o) refinement for 2-layered 3D designs.

Benchmark	# of TSV				# of buffers				Latency (ps)			
	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2
ispd09f11	36	36	36	36	130	138	131	131	405.3	390.8	405.6	405.6
ispd09f12	39	41	39	39	119	129	120	120	366.7	382.6	389.8	389.8
ispd09f21	36	38	36	36	129	137	129	129	410.9	457.1	420.0	420.0
ispd09f22	28	29	28	28	88	84	88	88	320.3	343.7	333.7	333.7
ispd09f31	78	89	78	78	286	283	286	286	611.1	614.4	615.8	615.8
ispd09f32	58	61	58	58	213	213	213	213	613.7	602.6	614.2	614.2
ispd09f33	63	69	69	69	212	226	216	216	555.3	549.4	568.8	568.8
ispd09f34	43	52	43	43	173	179	176	176	590.0	539.7	590.8	590.8
ispd09f35	55	62	55	55	200	199	201	201	549.7	559.7	553.3	553.3
Ratio	1.000	1.083	1.000	1.000	1.000	1.027	1.064	1.064	1.000	1.007	1.024	1.024

Table 3.2: Comparison of results by [5, 6] and LEE-CLK with (w/o) refinement for 4-layered 3D designs.

Benchmark	# of TSV				# of buffers				Latency (ps)			
	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2
ispd09f11	77	82	77	77	109	123	108	108	323.3	295.3	323.6	323.6
ispd09f12	69	75	69	69	109	105	110	110	317.9	317.8	318.3	318.3
ispd09f21	72	75	72	72	115	113	113	113	334.9	348.4	320.1	320.1
ispd09f22	60	61	60	60	73	83	73	73	268.6	266.6	268.9	268.9
ispd09f31	175	190	175	175	252	277	252	252	492.3	509.6	493.3	493.3
ispd09f32	129	150	129	129	184	209	184	184	465.1	462.3	467.5	467.7
ispd09f33	147	165	147	147	191	206	191	191	420.9	437.4	422.5	422.1
ispd09f34	107	106	107	107	148	161	148	148	430.9	415.3	427.3	427.3
ispd09f35	138	147	138	138	175	190	175	175	403.0	403.0	417.9	417.9
Ratio	1.000	1.071	1.000	1.000	1.00	1.078	1.000	1.00	1.000	0.998	1.000	1.000

Table 3.3: Comparison of mean and standard deviation of global skew in the clock trees synthesized by [5, 6] and LEE-CLK with (w/o) refinement for uncorrelated WID for 2-layered 3D designs.

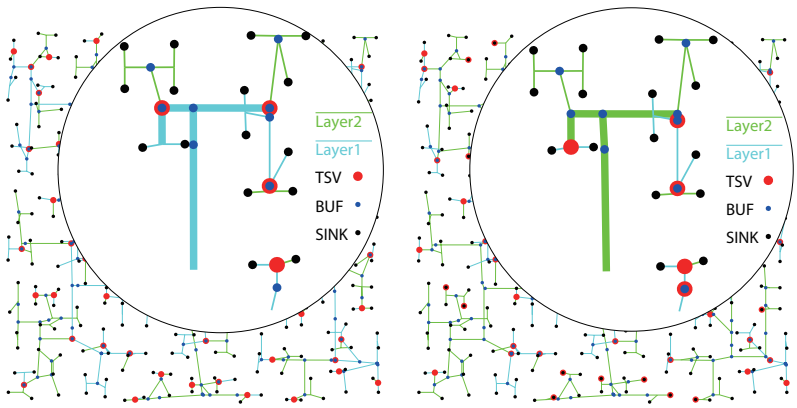
Benchmark	Global skew mean (ps)				Global skew std (ps)			
	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2
ispd09f11	49.54	64.23	48.04	48.16	9.55	11.11	9.04	9.09
ispd09f12	58.09	56.88	53.42	53.71	15.84	16.56	13.52	13.49
ispd09f21	66.4	67.78	63.28	63.03	21.3	24.18	18.17	18.11
ispd09f22	48.81	58.47	46.79	46.07	13.57	21.72	11.01	11
ispd09f31	66.58	77.14	64.28	64.29	12.53	19.82	12.46	12.41
ispd09f32	74.34	70.66	70.18	70.24	19.76	18.35	16.51	16.55
ispd09f33	72.13	87.61	66.08	66.87	16.65	33.98	13.24	13.3
ispd09f34	79.68	91.36	67.94	67.92	25.69	37.85	16.14	16.14
ispd09f35	83.73	81.45	69.61	69.34	27.8	27.97	17.18	17.08
Ratio	1.000	1.104	0.923	0.923	1.000	1.331	0.815	0.815

viation of the global skew significantly over the previous works [5, 6]. Tables 3.5 and 3.9 also show that LEE-CLK with refinement is more effective to reduces the mean and standard deviation of the global skew compared to LEE-CLK without refinement. Fig. 3.8 shows views of 3D clock tree for 2-layered design of *ispd09f33*, in which we can see that more edges are embedded in layer 2 by our method.

• **Comparing the parametric yield**: Tables 3.4, 3.6, 3.6 and 3.10 show comparisons of the parametric yields produced by the algorithm in [5, 6] and ours with the global skew bound 60ps, 80ps, and 100ps for 2-layered and 4-layered designs under both uncorrelated WID and spatially correlated WID, respectively. It is clearly revealed that our proposed on-package variation aware edge embedding algorithm is very effective in enhancing chip yield: 19.4∼44.4%, 12.3∼17%, 5.3∼8.7% improvement for the skew bound 60ps, 80ps, 100ps over that of the previous method. Fig. 3.9 depicts two global skew distributions for 2-layered design of *ispd09f35*, from which we can

Table 3.4: Comparison of yield by [5, 6] and LEE-CLK with (w/o) refinement for uncorrelated WID for 2-layered 3D designs.

Benchmark	Yield (%)											
	B_{skew} = 60 (ps)				B_{skew} = 80 (ps)				B_{skew} = 100 (ps)			
	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2
ispd09f11	87.1	51.5	90.8	90.4	99.6	80.5	100	100	100	92.8	100	100
ispd09f12	62.5	64.8	70.0	71.9	90.9	92.0	95.4	95.5	98.4	97.9	99.3	99.6
ispd09f21	45.8	46.8	53.7	52.8	78.3	73.2	83.4	84.8	92.9	89.4	95.6	95.4
ispd09f22	81.8	61.4	88.0	89.6	97.4	84.0	99.9	99.2	99.8	95.1	100	100
ispd09f31	35.0	18.7	40.3	41.1	85.7	63.2	91.3	90.3	98.6	87.9	98.5	98.6
ispd09f32	24.9	33.9	30.4	30.0	68.1	73.8	76.0	76.6	90.9	93.4	95.2	95.0
ispd09f33	26.5	20.1	33.0	33.3	72.9	52.3	85.4	84.6	84.0	72.7	99.0	99.1
ispd09f34	24.5	22.4	36.5	36.8	59.1	48.2	80.3	79.0	80.5	68.0	96.6	96.2
ispd09f35	19.7	24.0	35.7	34.5	55.4	56.2	76.1	76.6	77.4	80.0	95.1	95.0
Ratio	1.00	0.910	1.259	1.258	1.000	0.887	1.137	1.135	1.000	0.945	1.077	1.076



(a) (b)

Figure 3.8: Comparison of 3D clock trees for 2-layered circuit *ispd09f33*. (a) Result synthesized by [5]. (b) Result synthesized by LEE-CLK.

Table 3.5: Comparison of mean and standard deviation of global skew in the clock trees synthesized by [5, 6] and LEE-CLK with (w/o) refinement for uncorrelated WID for 4-layered 3D designs.

Benchmark	Global skew mean (ps)				Global skew std (ps)			
	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2
ispd09f11	49.58	49.8	46.04	45.7	10.6	12.4	9.84	8.88
ispd09f12	58.21	53.3	53.99	50.66	15.75	17.4	15.8	11.49
ispd09f21	64.22	55.68	56.0	58.14	18.79	15.8	15.23	14.88
ispd09f22	40.39	41.1	38.46	38.43	9.60	9.48	9.24	9.11
ispd09f31	86.09	63.56	83.31	73.07	29.94	13	27.37	18.61
ispd09f32	79.26	74.2	63.09	66.81	27.33	23.55	20.26	18.65
ispd09f33	70.88	68.24	63.68	64.46	19.77	16.92	16.14	16.88
ispd09f34	79.81	75.27	62.58	64.37	27.96	27.74	21.69	21.94
ispd09f35	66.10	61.4	61.0	59.56	16.96	15.42	14.92	13.41
Ratio	1.000	0.925	0.895	0.854	1.000	0.906	0.870	0.782

see that the distribution in Fig. 3.9(b) is much denser than that in Fig. 3.9(a).

3.5 Summary

This work addressed a new problem of layer embedding of clock edges in 3D clock tree synthesis with the objective of minimizing the impact of on-package variation on the clock skew. First, we showed that a careful clock edge embedding can greatly reduce the impact of on-package variation on the 3D clock skew, thus enhancing chip yield. We formulated the on-package variation aware edge embedding problem into a problem of maximizing the sharing of layers among the clock paths to minimize the impact of on-package variation globally and solved it efficiently, followed by applying a fine-grained refinement technique to balance the clock latency locally among the clock paths. From the experiments with benchmark circuits, it was shown that the

Table 3.6: Comparison of yield by [5, 6] and LEE-CLK with (w/o) refinement for uncorrelated WID for 4-layered 3D designs.

Benchmark	Yield (%)											
	B_{skew} = 60 (ps)				B_{skew} = 80 (ps)				B_{skew} = 100 (ps)			
	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2
ispd09f11	85.0	83.1	91.8	93.0	89.0	99.1	99.7	95.0	100.0	99.9	100.0	100.0
ispd09f12	60.0	71.1	70.0	79.0	90.0	94.4	92.4	100.0	99.0	98.5	99.3	100.0
ispd09f21	45.0	68.6	66.7	59.0	83.0	92.9	93.4	94.0	96.0	98.7	98.9	98.0
ispd09f22	96.0	96.9	98.0	99.0	99.0	100.0	99.9	100.0	100.0	100.0	100.0	100.0
ispd09f31	14.0	45.9	20.3	29.0	54.0	88.8	54.3	68.0	72.0	99.1	77.3	93.0
ispd09f32	28.0	33.5	43.4	45.0	58.0	68.3	76.0	79.0	78.0	86.5	93.0	93.0
ispd09f33	33.0	35.3	48.0	44.0	78.0	80.1	85.4	85.0	92.0	95.7	97.7	94.0
ispd09f34	26.0	33.9	56.5	49.0	58.0	65.2	81.3	78.0	78.0	83.2	93.6	94.0
ispd09f35	43.0	53.2	51.7	58.0	80.0	89.3	89.1	92.0	95.0	98.0	99.1	99.0
Ratio	1.000	1.420	1.398	1.444	1.000	1.153	1.134	1.170	1.000	1.072	1.067	1.087

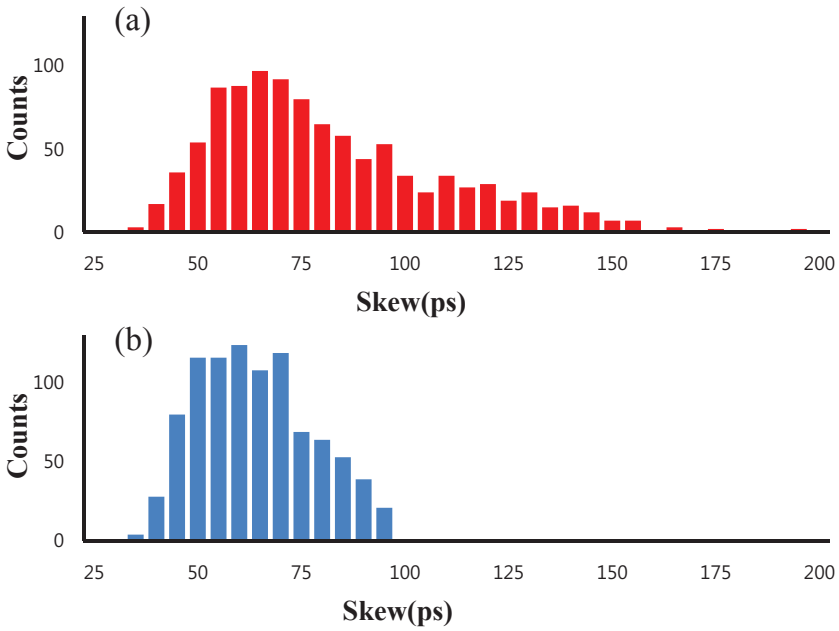


Figure 3.9: Comparison of the global skew distribution for 2-layered design of circuit *ispd09f35*. (a) Result synthesized by [5]. (b) Result synthesized by LEE-CLK.

Table 3.7: Comparison of mean and standard deviation of global skew in the clock trees synthesized by [5,6] and LEE-CLK with (w/o) refinement for spatially correlated WID for 2-layered 3D designs.

Benchmark	Global skew mean (ps)				Global skew std (ps)			
	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2
ispd09f11	40.2	45.51	34.8	34.84	14.73	21.83	12.27	12.58
ispd09f12	42.69	43.63	39.56	39.6	18.32	15.34	15.15	15.22
ispd09f21	52.43	63.17	44.45	44.58	24.52	32.35	17.4	17.34
ispd09f22	31.96	38.66	27.14	27.1	13.35	18.2	10.04	10.03
ispd09f31	68.49	86.01	59.65	59.6	28.48	45.71	22.71	22.73
ispd09f32	61.13	56.06	55.81	55.85	23.77	24.63	21.19	21.17
ispd09f33	57.34	62.04	49.3	49.4	21.58	25.09	17.18	17.23
ispd09f34	65.73	78.67	50.94	50.96	31.3	42.39	17.07	17.11
ispd09f35	68.68	61.62	50.42	50.35	32.76	26.75	18.89	18.82
Ratio	1.000	1.102	0.849	0.850	1.000	1.220	0.748	0.750

proposed solution of the layer embedding of edges was able to enhance the chip yield by 6.2∼25.8% and 5.3∼44.4% for 2-layered and 4-layered 3D designs, respectively.

Table 3.8: Comparison of yield by [5, 6] and LEE-CLK with (w/o) refinement for spatially correlated WID for 2-layered 3D designs.

Benchmark	Yield (%)											
	B_{skew} = 60 (ps)				B_{skew} = 80 (ps)				B_{skew} = 100 (ps)			
	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2
ispd09f11	90.8	78.0	95.9	95.8	98.6	91.8	99.6	99.6	99.8	97.7	99.9	99.9
ispd09f12	84.2	86.8	90.8	90.8	96.1	98.3	98.6	98.7	99.1	99.7	99.8	99.7
ispd09f21	67.4	55.1	82.6	82.1	86.8	73.7	96.0	96.0	95.3	87.7	99.7	99.7
ispd09f22	96.5	87.4	99.4	99.4	99.5	97.3	100.0	100.0	100.0	99.6	100.0	100.0
ispd09f31	44.9	35.9	55.5	55.6	69.2	54.5	80.9	81.1	86.2	68.1	95.8	95.7
ispd09f32	55.3	64.2	63.4	63.5	80.8	84.7	88.2	88.2	93.5	94.1	95.8	95.8
ispd09f33	62.8	53.9	77.3	77.2	85.7	79.4	94.0	94.1	95.5	91.4	99.7	99.5
ispd09f34	52.8	40.5	74.9	75.2	74.1	61.2	93.4	93.5	85.9	74.0	98.9	98.9
ispd09f35	49.5	55.0	73.6	73.8	68.5	78.6	91.7	91.6	83.9	91.7	98.6	98.6
Ratio	1.000	0.923	1.212	1.213	1.000	0.946	1.123	1.123	1.000	0.957	1.062	1.062

Table 3.9: Comparison of mean and standard deviation of global skew in the clock trees synthesized by [5,6] and LEE-CLK with(w/o) refinement for spatially correlated WID for 4-layered 3D designs.

Benchmark	Global skew mean (ps)				Global skew std (ps)			
	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2
ispd09f11	34.33	35.17	32.66	32.62	11.21	12.17	10.98	11.03
ispd09f12	41.78	37.21	41.56	37.87	16.19	16.42	16.03	13.56
ispd09f21	48.23	41.69	41.4	41.35	23.36	14.35	17.14	17.19
ispd09f22	25.21	28.54	24.56	24.64	8.35	9.74	8.3	8.29
ispd09f31	68.89	43.9	68.5	53.82	30.1	15.18	30.66	18.71
ispd09f32	67.39	60.88	51.7	52.46	32.7	27.45	21.41	21.86
ispd09f33	52.74	54.16	48.17	46.36	19.85	16.2	17.14	16.58
ispd09f34	64.34	61.42	51.56	51.28	32.96	31.09	25.5	25.68
ispd09f35	53.16	45.6	47.84	45.26	18.81	17.09	17.52	17
Ratio	1.000	0.921	0.906	0.864	1.000	0.877	0.882	0.818

Table 3.10: Comparison of yield by [5, 6] and LEE-CLK with (w/o) refinement for spatially correlated WID for 4-layered 3D designs.

Benchmark	Yield (%)											
	B_{skew} = 60 (ps)				B_{skew} = 80 (ps)				B_{skew} = 100 (ps)			
	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2	[5]	[6]	LEE-CLK w/o 3.3.2	LEE-CLK with 3.3.2
ispd09f11	98.1	96.8	97.8	97.8	100.0	99.8	100.0	100.0	100.0	100.0	100.0	100.0
ispd09f12	87.7	90.9	88.0	94.0	97.8	98.3	97.4	99.9	99.6	99.7	99.8	99.9
ispd09f21	73.5	89.6	86.1	86	89.5	98.4	96.7	99.4	97.0	99.8	99.4	99.4
ispd09f22	100.0	99.6	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
ispd09f31	47.4	85.2	47.3	68.0	69.4	98.0	69.4	98.1	85.2	99.6	84.8	98.1
ispd09f32	48.8	59.3	69.3	69.5	68.9	78.3	90.1	96.9	83.8	90.5	97.5	96.9
ispd09f33	67.8	70.8	76.4	80.5	91.2	92.4	96.1	99.4	98.4	99.0	99.4	99.4
ispd09f34	54.6	58.3	69.0	69.3	71.9	76.3	85.7	95.0	84.8	87.6	95.3	95.0
ispd09f35	68.4	81.7	76.9	81.3	91.5	95.4	95.6	99.5	98.2	99.2	98.8	99.5
Ratio	1.000	1.173	1.123	1.194	1.000	1.085	1.075	1.161	1.000	1.036	1.036	1.053

Chapter 4

Conclusion

Throughout this dissertation, we discussed about on-package variation and its impact on 3D IC design, and proposed algorithms that handle these design consideration about on-package variation during clock tree synthesis and post-silicon tuning design phase. The contributions of this dissertation are summarized as follows:

4.1 Chapter 2

In Chapter 2, we presented comprehensive die-to-die/wafer-to-wafer matching algorithms to improve the parametric yield of 3D ICs. The key part is to take into account the effect of current two-dimensional die/wafer matching on the result of subsequent two-dimensional die/wafer matching sequence, so that a globally maximal multi-dimensional matching can be achieved. Specifically, we formulated the two-dimensional die/wafer matching problem into a problem of finding a maximum flow of minimum cost in a network and solved it optimally in polynomial time. From the experimental results using benchmark circuits, it was shown that the proposed algorithm was able to improve the parametric yield by 2%, 7%, and 8% for 3-layered, 4-layered,

and 5-layered 3D ICs, respectively.

4.2 Chapter 3

We addressed a new problem of layer embedding of clock edges in 3D clock tree synthesis with the objective of minimizing the impact of on-package variation on the clock skew. First, we showed that a careful clock edge embedding can greatly reduce the impact of on-package variation on the 3D clock skew, thus enhancing chip yield. Then, we formulated the on-package variation aware edge embedding problem into a problem of maximizing the sharing of layers among the clock paths to minimize the impact of on-package variation globally and solved it efficiently, followed by applying a fine-grained refinement technique to balance the clock latency locally among the clock paths. From the experiments with benchmark circuits, it was shown that the proposed solution of the layer embedding of edges was able to enhance the chip yield by 6.2$\sim$25.8% and 5.3$\sim$44.4% for 2-layered and 4-layered 3D designs, respectively.

Bibliography

[1] S. Reda, A. Si, and R. I. Bahar, "Reducing the leakage and timing variability of 2d ics using 3d ics," in *Proceedings of the 15th ACM/IEEE International Symposium on Low Power Electronics and Design*, Aug. 2009, pp. 283–286.

[2] K. Chae, X. Zhao, S. K. Lim, and S. Mukhopadhyay, "Tier adaptive body biasing: A post-silicon tuning method to minimize clock skew variations in 3-d ics," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 3, no. 10, pp. 1720–1730, Oct. 2013.

[3] T.-Y. Kim and T. Kim, "Post silicon management of on-package variation induced 3d clock skew," *Journal of Semiconductor Technology and Science*, vol. 12, no. 2, pp. 139–149, 2012.

[4] S. Reda, G. Smith, and L. Smith, "Maximizing the functional yield of wafer-to-wafer 3-d integration," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 17, no. 9, pp. 1358–1362, Sep. 2009.

[5] T.-Y. Kim and T. Kim, "Clock tree synthesis for tsv-based 3d ic designs," *ACM Transactions on Design Automation of Electronic Systems*, vol. 16, no. 4, pp. 48:1–48:21, Oct. 2011.

[6] X. Zhao, J. Minz, and S. K. Lim, "Low-power and reliable clock network design for through-silicon via (tsv) based 3d ics," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 1, no. 2, pp. 247–259, Feb. 2011.

[7] Y. Ye, S. Gummalla, C.-C. Wang, C. Chakrabarti, and Y. Cao, "Random variability modeling and its impact on scaled cmos circuits," *Journal of Computational Electronics*, vol. 9, no. 3-4, pp. 108–113, 2010. [Online]. Available: http://dx.doi.org/10.1007/s10825-010-0336-5

[8] H. Chang and S. S.Sapatnekar, "Statistical timing analysis considering spatial correlations using a single pert-like traversal," in *Proceedings of the 2003 IEEE/ACM international conference on Computer-aided design*. IEEE Computer Society, 2003, p. 621.

[9] J. V. Olmen, A. Mercha, G. Katti, C. Huyghebaert, J. V. Aelst, E. Seppala, Z. Chao, S. Armini, J. Vaes, R. C. Teixeira, M. V. Cauwenberghe, P. V. K. Verhemeldonck, A. Jourdain, W. Ruythooren, M. de Potter de ten Broeck, A. Opdebeeck, T. Chiarella, B. Parvais, I. Debusschere, T. Hoffmann, B. D. Wachter, W. Dehaene1, M. Stucchi, M. Rakowski, P. Soussan, R. Cartuyvels, and E. Beyne, "3d stacked ic demonstration using a through silicon via first approach," in *Proceedings of the 2008 IEEE International Electron Devices Meeting*, 2008, pp. 1–4.

[10] D. H. Kim, K. Athikulwongse, and S. K. Lim, "A study of through-silicon-via impact on the 3d stacked ic layout," in *Proceedings of the 2009 International Conference on Computer-Aided Design*, 2009, pp. 674–680.

[11] F. Akopyan, C. T. O. Otero, D. Fang, , S. J. Jackson, and R. Manohar, "Variability in 3-d integrated circuits," in *Proceedings of the 2008 IEEE Custom Integrated Circuits Conference,*, 2008, pp. 659–662.

[12] S. Garg and D. Marculescu, "3d-gcp: An analytical model for the impact of process variations on the critical path delay distribution of 3d ics," in *Proceedings of the 10th IEEE International Symposium on Quality Electronic Design*, Mar. 2009, pp. 147–155.

[13] S. Ozdemir and et al., "Quantifying and coping with parametric variations in 3d-stacked microarchitectures," in *Proceedings of the 47th ACM/IEEE Design Automation Conference*, Jun. 2010, pp. 144–149.

[14] H. Xu, V. F. Pavlidis, and G. D. Micheli, "Effect of process variations in 3d global clock distribution networks," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 8, no. 3, p. 20, Aug. 2012.

[15] J.-S. Yang, J. Pak, X. Zhao, S. K. Lim, and D. Z. Pan, "Robust clock tree synthesis with timing yield optimization for 3d-ics," in *Proceedings of the 16th IEEE Asia and South Pacific Design Automation Conference*, Jan. 2011, pp. 621–626.

[16] C. Ferri, S. Reda, and R. I. Bahar, "Parametric yield management of 3d ics: models and strategies for improvement," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 4, no. 4, pp. 19:1–19:22, Oct. 2008.

[17] K. Chae and S. Mukhopadhyay, "Tier-adaptive-voltage-scaling (tavs): A methodology for post-silicon tuning of 3d ics," in *Proceedings of the 2012 17th Asia and South Pacific Design Automation Conference*, Jan. 2012, pp. 277–282.

[18] J. Kim, D. Joo, and T. Kim, "An optimal algorithm of adjustable delay buffer insertion for solving clock skew variation problem," in *Proceedings of the 2013 50th ACM/EDAC/IEEE Design Automation Conference*, Jun. 2013, pp. 1–6.

[19] T.-H. Chao, Y.-C. Hsu, J.-M. Ho, K. D. Boese, and A. B. Kahng, "Zero-skew clock routing with minimum wirelength," *IEEE Transactions on Circuits and Systems*, vol. 39, no. 11, pp. 799–814, Nov. 1992.

[20] J. Cong, A. B. Kahng, C.-K. Koh, and C.-W. A. Tsao, "Bounded-skew clock and steiner routing," *ACM Transactions on Design Automation of Electronic Systems*, vol. 3, pp. 341–388, Jul. 1998.

[21] B. Lu, J. Hu, G. Ellis, and H. Su, "Process variation aware clock tree routing," in *Proceedings of the 2003 international symposium on Physical design*, 2003, pp. 174–181.

[22] U. Padmanabhan, J. M. Wang, and J. Hu, "Robust clock tree routing in the presence of process variations," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 27, no. 8, pp. 1385–1397, Aug. 2008.

[23] M. R. Guthaus, D. Sylvester, and R. B. Brown, "Process-induced skew reduction in nominal zero-skew clock trees," in *Proceedings of the 2006 11st Asia and South Pacific Design Automation Conference*, 2006, pp. 84–89.

[24] G. Smith, L. Smith, S. Hosali, and S. Arkalgud, "Yield consideration in the choice of 3d technology," in *Proceedings of the 2007 International Symposium on Semiconductor Manufacturing*, Oct. 2007, pp. 1–3.

[25] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network flows: theory, algorithms, and applications*. Prentice hall, 1993.

[26] A. V. Goldberg and R. E. Tarjan, "Finding minimum-cost circulations by canceling negative cycles," *Journal of the ACM*, vol. 36, no. 4, pp. 873–886, 1989.

[27] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistic Quarterly*, vol. 2, pp. 83–97, 1955.

[28] ISPD. (2009) ISPD 2009 clock network synthesis contest. [Online]. Available: http://ispd.cc/contests/09/ispd09cts.html

[29] PTM. (2011) Predictive technology model. [Online]. Available: http://ptm.asu.edu/

[30] X. Zhao, D. L. Lewis, H.-H. S. Lee, and S. K. Lim, "Low-power clock tree design for pre-bond testing of 3-d stacked ics," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 5, pp. 732–745, May 2011.

[31] T.-Y. Kim and T. Kim, "Resource allocation and design techniques of prebond testable 3-d clock tree," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 1, pp. 138–151, Jan. 2013.

[32] W. Liu, H. Du, Y. Wang, Y. Ma, Y. Xie, J. Quan, , and H. Yang, "Tsv-aware topology generation for 3d clock tree synthesis," in *Proceedings of the 2013 IEEE International Symposium on Quality Electronic Design*, Mar. 2013, pp. 300–307.

[33] L. R. Ford and D. R. Fulkerson, *Flows in Networks*. Princeton University Press, Princeton, 1962.

[34] E. Dahlhaus, D. S. Johnson, and C. H. Papadimitrious, *The complexity of multiway cuts*. Symposium on Theory of Computing, 1992.

[35] B. W. Kernighan and S. Lin, "An efficient heuristic procedure for partitioning graphs," *Bell System Technical Journal*, vol. 49, no. 2, pp. 291–307, 1970.

[36] R. Chaturvedi and J. Hu, "Buffered clock tree for high quality ic design," in *Proceedings of the 5th IEEE International Symposium on Quality Electronic Design*, Mar. 2004, pp. 381–386.

[37] A. Agarwal, D. Blaauw, , and V. Zolotov, “Statistical timing analysis for intra-die process variations with spatial correlations,” in *Proceedings of the 2003 ACM/IEEE International Conference on Computer-Aided Design*, Nov. 2003, pp. 900–07.

국문 초록

CMOS의 크기가 줄어드는 것에 따라, 속도와 파워 같은 칩 수행 계수의 변이를 제어하는 것이 칩 수율을 높이기 위해 아주 중요해지고 있다. 증가 되고 있는 칩 수행 계수의 변이는 가드 밴드 증가와 디자인 터어라운드 시간 증가의 같은 추가적인 노력을 요구하게 되고, 이것은 칩 수행 능력과 경제적 이익을 저하시킨다. 한편, 실리콘 관통 비아 (TSV)를 기반으로 하는 3차원 직접회로 기술은 긴 연결선 과 거대한 다이 크기 문제에 대한 유력한 해결책으로 여겨지고 있다. 3차원 집적 회로는 다른 웨이퍼에서 제작된 여러 다이들을 쌓아서 생산되기 때문에, 아주 다른 공정 특성을 가진 다이의 결합은 같은 칩내에서도 소자들의 특성 차이를 확대 시킬 수도 있다. 본 논문에서는 3차원 집적회로내의 변이를 분석하고 이를 완화할 수 있는 효과적인 방법을을 제시한다. 처음으로, 3차원 집적회로를 이루는 다이를 선택하는 방법을 이용하여 클락 스큐의 변동성을 완화하는 방법을 제안한다. 구체적으로 다층 구조를 가진 3D IC에 대해서 포스트 실리콘 조정 기술을 고려하여 클락 스큐를 최소화 시키는 3D 결합 방법이 제시 되었다. 두번째로, 세심한 클락 에지 할당이 3차원 클락 스큐에 대한 패키지내 변동성을 줄일 수 있다는 것을 보여주고, 3차원 클락 합성에서 패키지내 변동성을 고려한 레이어 할당 문제에 대한 두 단계 해결책을 제시한다. 요약하면, 본 논문은 공정 변이에 내성있는 3차원 집적회로를 위한 효과적인 3차원 결합 방법과 3차원 클락 합성 방법을 제시한다.

주요어: VLSI&CAD, 3차원 집적회로, 실리콘 관통 비아, 클락 트리 합성, 최적화

학번: 2009-30915